Technical Note

1967-48

A. Bertolini

A Trajectory Analysis Program (TRAP)

29 September 1967

Prepared for the Advanced Research Projects Agency
under Electronic Systems Division Contract AF 19(628)-5167 by

Lincoln Laboratory
MASSACHUSETTS INSTITUTE OF TECHNOLOGY
Lexington, Massachusetts

The work reported in this document was performed at Lincoln Laboratory, a center for research operated by Massachusetts Institute of Technology. This research is a part of Project DEFENDER, which is sponsored by the U.S. Advanced Research Projects Agency of the Department of Defense; it is supported by ARPA under Air Force Contract AF 19(628)-5167 (ARPA Order 498).

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

LINCOLN LABORATORY

A TRAJECTORY ANALYSIS PROGRAM (TRAP)

A. BERTOLINI

Group 42

TECHNICAL NOTE 1967-48

29 SEPTEMBER 1967

LEXINGTON MASSACHUSETTS

ABSTRACT

A FORTRAN IV program package has been written for the generation, analysis, and estimation of re-entry trajectories. The following functions may be performed by the program:

(1) Generation of simulated noiseless trajectory data by integrating the differential equations of motion, using a predictor-corrector method.

(2) Error analyses on trajectories generated by the program, in which the effects of hypothetical errors in the initial state and subsequent measurements may be studied.

(3) Estimation of state vector quantities, using a recursive Kalman-filter scheme, from

   (a) Simulated noisy data generated by the program.

   (b) Noisy radar measurements, supplied externally to the program.

Accepted for the Air Force
Franklin C. Hudson
Chief, Lincoln Laboratory Office

CONTENTS

A TRAJECTORY ANALYSIS PROGRAM (TRAP)

I. INTRODUCTION

A FORTRAN IV double-precision program package has been written for the generation, analysis, and estimation of re-entry trajectories. The trajectories are characterized by a seven-component state vector specifying position, first time derivative of position, and drag parameter defined as the reciprocal of the weight-to-drag ratio. The program assumes a spherical, rotating earth with an atmosphere and uses either English or metric units. The following functions may be performed by the program package:

(a) Generation of simulated noiseless trajectory data by integrating the differential equations of motion, using a predictor-corrector method.

(b) Error analyses on trajectories generated by the program, in which the effects of hypothetical errors in the initial state and subsequent measurements may be studied.

(c) Estimation of state vector quantities, using a recursive Kalman-filter scheme, from

  (1) Simulated noisy data generated by the program.

  (2) Noisy radar measurements, supplied externally to the program.

A modified version of the main program is required to perform function (c) (2).

The state vector which describes the trajectory motion has seven components, and may be considered in two coordinate systems. In radar-centered rectangular coordinates, with x pointing east, y north, and z vertically upward, we have the following

$x_1 = x$

$x_2 = y$

$x_3 = z$

$x_4 = \dot{x}$

$x_5 = \dot{y}$

$x_6 = \dot{z}$

$x_7 = \alpha$ = drag parameter = $1/\beta$

In radar-centered polar coordinates, which is the usual measurement coordinate system

r_1 = range

r_2 = azimuth

r_3 = elevation

r_4 = range rate

r_5 = azimuth rate

r_6 = elevation rate

$r_7 = \alpha$ = drag parameter

Input and output quantities may be given in polar or rectangular coordinates but all calculations are done in rectangular coordinates. English (pounds, feet, seconds) or metric (meters, kilograms, seconds) units may be specified.

The program package consists of the main program plus ten subroutines. The subroutines perform various calculations, ranging from integrating the differential equations of motion to multiplying matrices. Some of these subroutines may be of general interest for use outside of the program package. The functions of the main program and all the subroutines are summarized in Table I, with full details given in later sections.

II. MAIN PROGRAM (VERSION I)

The main program reads input cards, calls the appropriate subroutines to perform the desired calculations, and prints output data. Data on the input cards completely specifies the program function (e. g., trajectory generation, error analysis, or estimation), furnishes required initial conditions, and specifies the amount of printed output desired. If required, the user may change the program so that the output data may be written on tape, plotted, or punched on cards. The formats of the seven required input cards are given in Table II. Any number of trajectories may be run in succession by stacking the input cards for each case in succession. A block diagram of the main program is shown in Fig. 1.

The following COMMON statements are used:

```
COMMON/ACOM/COVAR(7), SIGMA(7)/FCOM/COORD, DLAT,
PRNT/ICOM/KLAMP, MDATA, NN
```

An explanation of these variables is given in Table II.

Varying amounts of printed output may be obtained by proper specification of the print-selector parameter PRNT. These are summarized below:

PRNT < 0. Every 50 data points, the following tabulations are printed for the preceding 50 data points.

(a) Time, height, and the seven components of the nominal state vector in radar-centered polar coordinates.

(b) Time, aspect angle, and the seven components of the nominal state vector in radar-centered rectangular coordinates.

(c) Time, plus the measurement vector in radar-centered polar coordinates (printed only in estimation cases).

(d) Time, and the estimated state vector in radar-centered polar and rectangular coordinates (printed only in estimation cases).

(e) Time, and the errors (difference between estimated and nominal state vector) in polar and rectangular coordinates.

(f) Theoretical rms errors, obtained from covariance matrices, in polar and rectangular coordinates.

Note: The above may be printed out at data intervals other than every 50 points by properly specifying the value of IMAX in statement 9 of the main program (see Tables B-I and B-II in Appendix B). All one-dimensioned arrays of size 54 should be redimensioned to be equal to or greater than the new value of IMAX.

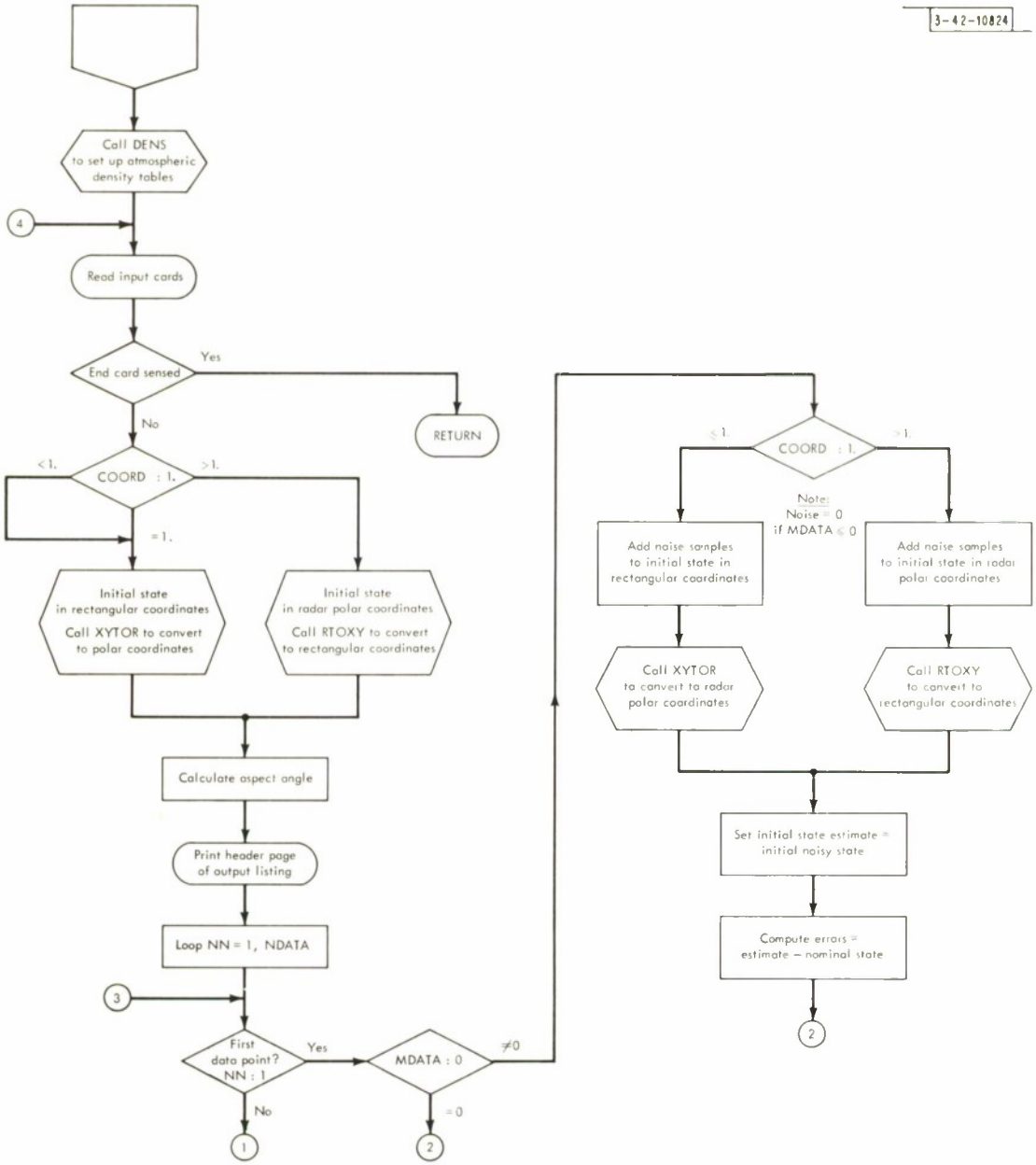

Fig. 1. Flow chart of TRAP (version I) main program.

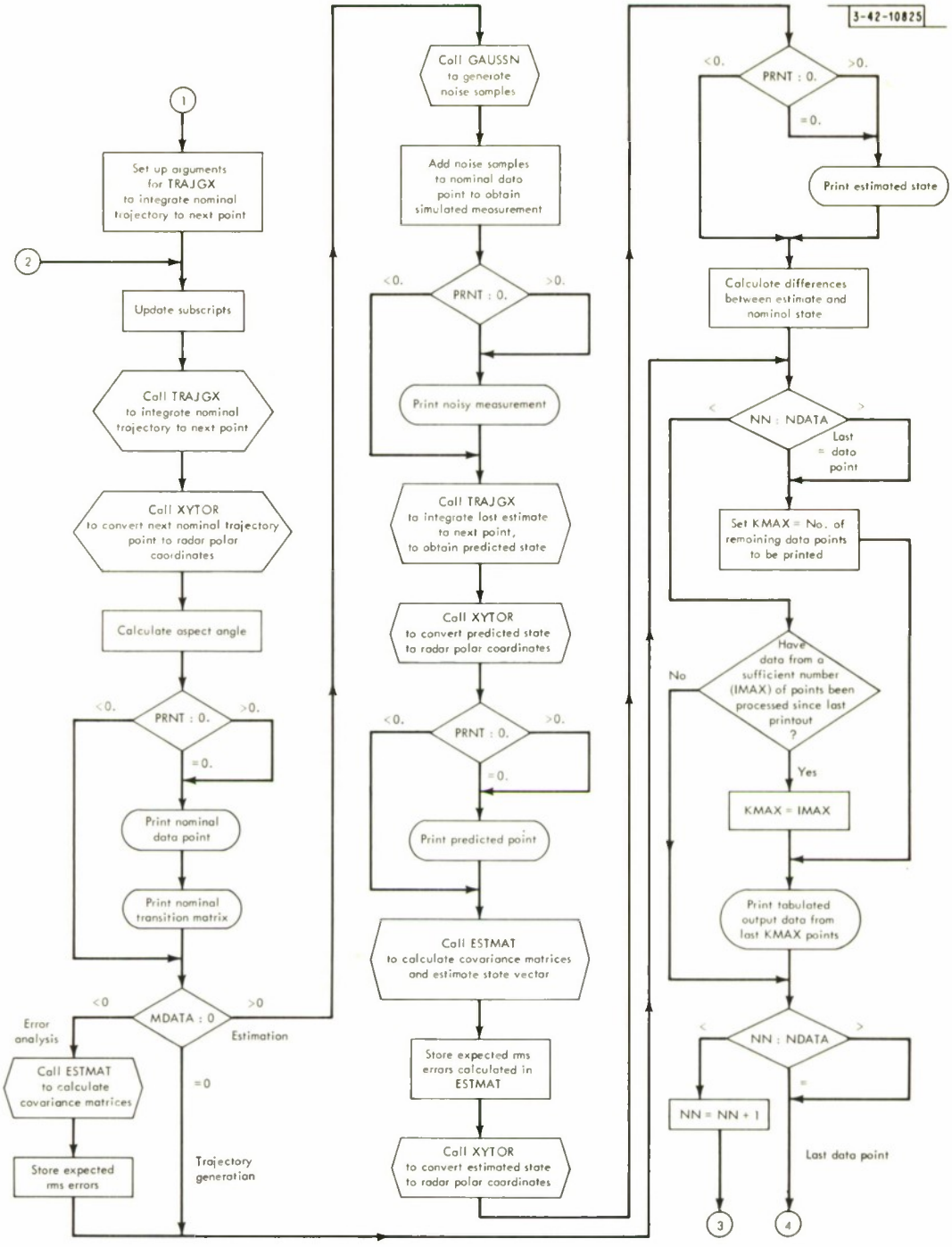

Fig. 1. Continued.

TABLE I

LIST OF PROGRAMS IN TRAP PACKAGE

Program or Subroutine	Function
Main Program (Versions I and II)	Reads input cards specifying desired calculations and initial conditions, calls appropriate subroutines, and prints out results.
TRAJGX	Integrates differential equations of motion in radar-centered rectangular coordinates. A spherical rotating earth with a realistic atmosphere is used.
ESTMAT	Estimates the true value of a state vector by combining a predicted value of the state vector with a noisy measurement. Also calculates covariance matrices for error analyses.
DENS	Calculates atmospheric density at any given altitude.
XYTOR	Converts from radar-centered rectangular coordinates to polar coordinates.
RTOXY	Converts from radar-centered polar coordinates to rectangular coordinates.
XCOVTR	Converts a mean-square error covariance matrix in rectangular coordinates to one in polar coordinates. It also calculates a partial derivative matrix $\underline{H} = \partial \underline{r} / \partial \underline{x}$ where $\underline{r}$ and $\underline{x}$ are the state vector in radar-centered polar and rectangular coordinates respectively.
RCOVTX	Converts a mean-square error covariance matrix in polar coordinates to one in rectangular coordinates.
GAUSSN	Generates random numbers with a Gaussian distribution.
DMTMUL	Multiplies two matrices in double precision.
MINV	Matrix inversion subroutine from IBM 360 Scientific Subroutine Package (double-precision version used).

PRNT = 0. All of the above are printed, as well as the following which are printed for every data point.

(a) Time, height, aspect angle, and the nominal state vector in both polar and rectangular coordinates.

(b) The transition matrix, $\partial \underline{x}(t_k)/\partial \underline{x}(t_{k-1})$ (see discussion of subroutine TRAJGX, Sec. IV), along the nominal trajectory in rectangular coordinates.

(c) The noisy measurement vector, in polar coordinates (printed only in estimation cases).

(d) The predicted state vector, in polar and rectangular coordinates (printed only in estimation cases).

(e) The covariance matrices of the estimated state vector (see discussion of subroutine ESTMAT, Sec. V).

(f) The estimated state vector in polar and rectangular coordinates.

PRNT > 0. All of the above are printed, as well as the following which are printed by subroutine ESTMAT for every data point.

(a) Transition matrix used in the estimation.

(b) Covariance matrices for predicted and estimated data points.

(c) Various vectors and matrices used in the calculation of the final covariance matrices and the estimate (see description of subroutine ESTMAT, Sec. V).

TABLE II

INPUT CARDS TO MAIN PROGRAM (VERSION I)

Card	Format	Arguments and/or Description
1	18A4	Title card with 72 alpha-numerical characters available.
2	7F10.3	(STATEI (J), J = 1, 7) where STATEI is initial value of nominal state vector, in either polar or rectangular radar-centered coordinates.
3	5F10.3, F5.3, 3I5	TZERO = initial time (in seconds) DELT = time of first measurement with respect to time of initial state TINT = interval, in seconds, between data points TINCR = trajectory integration step size, in seconds DLAT = latitude of radar (reference) site in degrees PRNT = print-selector parameter (see text, p. 2) NDATA = number of data points to be processed KLAMP = "memory" of estimation algorithm, given in number of data points (see description of ESTMAT, Sec. V); KLAMP = 0 is used to indicate infinite memory MDATA = mode parameter MDATA < 0 specifies error analysis MDATA = 0 specifies noiseless trajectory generation MDATA > 0 specifies estimation The magnitude of MDATA ⩽ 7 and specifies the size of the measurement vector used in calculations (see description of ESTMAT, Sec. V).
4	7F10.3	(SIGMA(J), J = 1, 6) = rms noise levels associated with measurement vector. No value is given to SIGMA(7) since the drag parameter, which is the seventh component of the state vector, is not measurable directly. COORD = Coordinate – specification parameter COORD = −2. Initial state vector given in radar-centered polar coordinates. Metric units used for all calculations. COORD = −1. Initial state vector given in radar-centered rectangular coordinates. Metric units used for all calculations. COORD = 1. Initial state vector given in radar-centered rectangular coordinates. English units used for all calculations. COORD = 2. Initial state vector given in radar-centered polar coordinates. English units used for all calculations.
5	7F10.3	(COVAR(J), J = 1, 7) = rms noise levels associated with initial state vector in units specified by the parameter COORD.
6	7F10.3	(FNOISE(J), J = 1, 7) = noise samples associated with initial state vector, in units specified by the parameter COORD.
7	7F10.3	(ERRMAX(J), J = 1, 7) = magnitude of maximum allowable convergence errors. The convergence errors are an indication of the accuracy of the integration of the trajectory equations – the smaller the errors, the better the accuracy. Values of 0.1 for all seven elements in the ERRMAX array have been found suitable for most cases. See the mathematical discussion of subroutine TRAJGX, Sec. IV, for a description of the convergence errors.

III. MAIN PROGRAM (VERSION II)

A second main program is available for use in estimating trajectory parameters from real radar measurement data. The measurement data are read in on cards but it is a simple matter to change the input medium and/or format. No error analysis or simulated trajectory generation is done. Four input cards, followed by the measurement data cards, are required (see Table III).

TABLE III
INPUT DATA CARDS TO MAIN PROGRAM (VERSION II)

Card	Format	Argument and/or Description
1	18A4	Title card, with 72 alpha-numerical characters available.
2	4F10.3, F5.3, 3I5	TZERO, TINT, TINCR, DLAT, PRNT, NDATA, KLAMP, MDATA (see text below).
3	7F10.3	(SIGMA(J), J = 1, 6), COORD
4	7F10.3	(ERRMAX(J), J = 1, 7) convergence errors
5 . .	D15.2, 4D15.6	Data cards for each measurement, giving time, range, azimuth, elevation, range rate

The title card and the arguments TINCR, DLAT, PRNT, NDATA, KLAMP, SIGMA, and ERRMAX are exactly as described previously for Version I. The arguments MDATA and COORD are also as described previously except that MDATA is limited to positive values while COORD must equal ±2.

The arguments TZERO and TINT specify the starting time of the desired data and the minimum desired data interval, respectively. All data prior to time TZERO are ignored, as well as succeeding data points which are less than TINT seconds later than the previously processed point.

It should be noted that an initial state vector to start the program is obtained from the measurements at time TZERO or later. Any required time derivatives not contained in the measurement are calculated by taking differences of measurements TINT seconds, or more, apart.

A block diagram of this program is given in Fig. 2.

IV. SUBROUTINE TRAJGX

A. Description

Subroutine TRAJGX is a double-precision FORTRAN subroutine which integrates the differential equations of motion in radar-centered rectangular coordinates over any specified time interval, using a predictor-corrector method. It assumes a spherical rotating earth with a rigid atmosphere. A choice of English units (feet, pounds, seconds), or metric units (meters, kilograms, seconds), is available by specifying the parameter COORD, in COMMON storage. Atmospheric data are obtained by calling subroutine DENS via the entry point ATM, described in Sec. VI. Subroutine TRAJGX may be used to generate a ballistic trajectory or to predict the position and velocity of a re-entry vehicle at any given time, given the position, velocity, and ballistic coefficient at some other time.

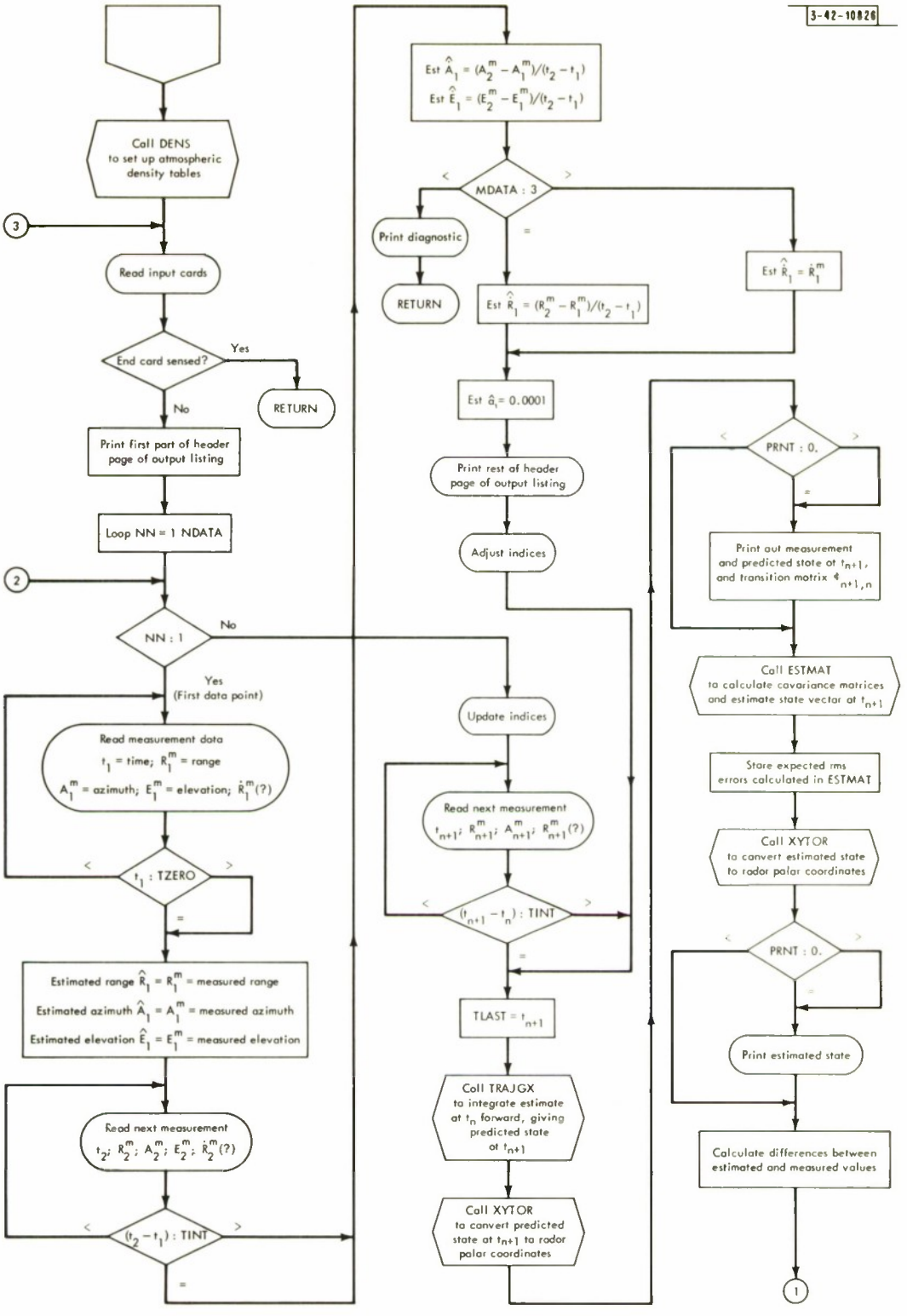

Fig. 2. Flow chart of TRAP (version II) main program.

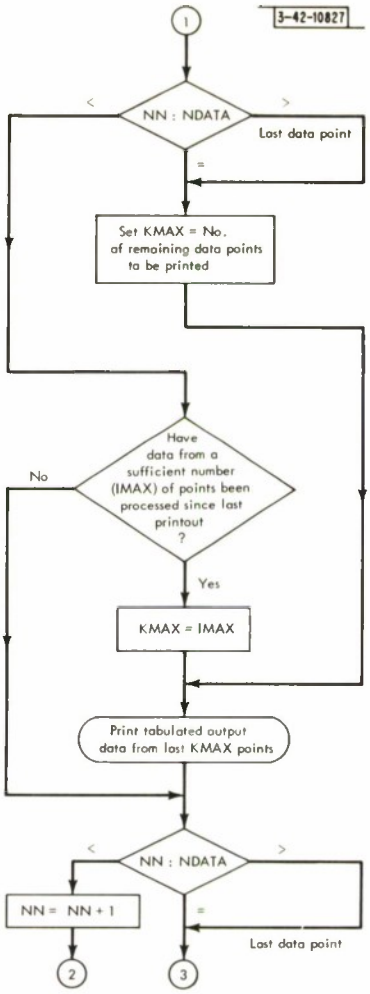

Fig. 2. Continued.

The subroutine is called by the following statement:

CALL TRAJGX (X1, Y1, Z1, XDOT1, YDOT1, ZDOT1, BETA1, TAU,
TSTEP, ERRMAX, X2, Y2, Z2, XDOT2, YDOT2, ZDOT2, BETA2, PHIMAT)

The input arguments are:

X1, Y1, Z1 } initial position in radar-centered rectangular coordinates

XDOT1 = $\dot{X}1$, YDOT1 = $\dot{Y}1$, ZDOT1 = $\dot{Z}1$ } initial velocity components

BETA1 = initial ballistic coefficient

TAU = integration time interval (sec)

TSTEP = integration step size (sec)

ERRMAX = convergence errors (see description below)

The output arguments are

X2, Y2, Z2 } position at time TAU seconds after initial time

XDOT2 = $\dot{X}2$, YDOT2 = $\dot{Y}2$, ZDOT2 = $\dot{Z}2$ } velocity components at time TAU seconds after initial time

BETA2 = ballistic coefficient at time TAU seconds after initial time. (At present, TRAJGX assumes that the ballistic coefficient is constant and sets BETA2 = BETA1. Provision can be made, however, for a varying ballistic coefficient).

PHIMAT = transition matrix, whose elements are defined by

$$\varphi_{ij} = \partial x_i(t_o + TAU)/\partial x_j(t_o)$$

where x_i is the ith component of a state vector $\underline{x}$, whose components are

$x_1(t_o)$ = X1

$x_2(t_o)$ = Y1

$x_3(t_o)$ = Z1

$x_4(t_o)$ = XDOT1

$x_5(t_o)$ = YDOT1

$x_6(t_o)$ = ZDOT1

$x_7(t_o)$ = 1/BETA1

Similarly

$$x_1(t_o + TAU) = X2$$

$$\vdots$$

$$x_7(t_o + TAU) = 1.0/BETA2$$

The following statements are also required in the calling program:

```
COMMON/FCOM/COORD, DLAT, PRNT/ICOM/KLAMP, MDATA, NN
DIMENSION ERRMAX(7)
```

where

DLAT = latitude, in degrees, of radar (reference) site

COORD > 0. indicates English system units of feet, pounds, seconds

COORD < 0. indicates metric system units of meters, kilograms, seconds

NN indicates the data point under current consideration (NN = 1 for the first data point and indicates that TRAJGX is being called for the first time. This is the only case of interest to TRAJGX)

ERRMAX is an array of convergence errors associated with the seven components of the state vector $\underline{x}$. In general, the smaller the specified convergence errors, the more accurate the integration, with a resulting increase in computation time. The significance of the convergence errors is discussed in Sec. IV. B. Values of 0.1 seem to be suitable for all seven convergence errors.

The following values have been used for some fundamental parameters.

$R_E = 2.0925738 \times 10^7$ ft (radius of the earth)

$G_M = 1.40763488 \times 10^{16}$ ft^3/sec^2 (Newton's gravitational constant x mass of earth)

$\omega = 7.27 \times 10^{-5}$ radians/sec (earth's angular rotation rate)

1 meter = 3.2808333 ft

$\pi = 3.14159265$

B. Mathematical Discussion

The program assumes the following basic equations of motion in radar-centered rectangular coordinates:[1]

$$\ddot{x} = P\alpha\dot{x} + Bx + C\dot{y} + E\dot{z}$$

$$\ddot{y} = -C\dot{x} + Gy + P\alpha\dot{y} + Jz + K$$

$$\ddot{z} = -E\dot{x} + Jy + Nz + P\alpha\dot{z} + Q$$

where

$\alpha = 1/\beta$ = reciprocal of ballistic coefficient = drag parameter

$P = -\rho V_D/2$ $\quad \begin{cases} \rho = \text{atmospheric density} \\ V_D = \text{drag velocity} = (\dot{x}^2 + \dot{y}^2 + \dot{z}^2)^{1/2} \end{cases}$

TABLE IV

$\underline{A}$ MATRIX = $\partial \underline{f}(\underline{x})/\partial \underline{x}$

$$\frac{\partial F}{\partial x} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ \left[B + x\partial B/\partial x + \dot{x}\left(\frac{\alpha \partial P}{\partial x} + P\frac{\partial \alpha}{\partial x}\right)\right] & \left[x\frac{\partial B}{\partial y} + \dot{x}\left(\alpha\frac{\partial P}{\partial y} + P\frac{\partial \alpha}{\partial y}\right)\right] & \left[x\frac{\partial B}{\partial z} + \dot{x}\left(\alpha\frac{\partial P}{\partial z} + P\frac{\partial \alpha}{\partial z}\right)\right] & \left(\alpha P + \alpha\dot{x}\frac{\partial P}{\partial \dot{x}}\right) & \left(\dot{x}\alpha\frac{\partial P}{\partial \dot{y}} + C\right) & \left(\dot{x}\alpha\frac{\partial P}{\partial \dot{z}} + E\right) & (P\dot{x}) \\ \left[y\frac{\partial G}{\partial x} + \dot{y}\left(P\frac{\partial \alpha}{\partial x} + \alpha\frac{\partial P}{\partial x}\right)\right] & \left[G + y\frac{\partial G}{\partial y} + \dot{y}\left(P\frac{\partial \alpha}{\partial y} + \alpha\frac{\partial P}{\partial y}\right)\right] & \left[y\frac{\partial G}{\partial z} + J + \dot{y}\left(P\frac{\partial \alpha}{\partial z} + \alpha\frac{\partial P}{\partial z}\right)\right] & \left(\dot{y}\alpha\frac{\partial P}{\partial \dot{x}} - C\right) & \left[\alpha\left(P + \dot{y}\frac{\partial P}{\partial \dot{y}}\right)\right] & \left(\dot{y}\alpha\frac{\partial P}{\partial \dot{z}}\right) & (P\dot{y}) \\ \left[\frac{\partial Q}{\partial x} + z\frac{\partial N}{\partial x} + \dot{z}\left(P\frac{\partial \alpha}{\partial x} + \alpha\frac{\partial P}{\partial x}\right)\right] & \left[J + \frac{\partial Q}{\partial y} + z\frac{\partial N}{\partial y} + \dot{z}\left(P\frac{\partial \alpha}{\partial y} + \alpha\frac{\partial P}{\partial y}\right)\right] & \left[N + z\frac{\partial N}{\partial z} + \frac{\partial Q}{\partial z} + \dot{z}\left(P\frac{\partial \alpha}{\partial z} + \alpha\frac{\partial P}{\partial z}\right)\right] & \left(\dot{z}\alpha\frac{\partial P}{\partial \dot{x}} - E\right) & \left(\dot{z}\alpha\frac{\partial P}{\partial \dot{y}}\right) & \left[\alpha\left(P + \dot{z}\frac{\partial P}{\partial \dot{z}}\right)\right] & (P\dot{z}) \\ \left(\frac{\partial \alpha}{\partial x}\right) \text{ (See Note 1)} & \left(\frac{\partial x}{\partial y}\right) \text{ (See Note 1)} & \left(\frac{\partial \alpha}{\partial z}\right) \text{ (See Note 1)} & 0 & 0 & 0 & 0 \end{bmatrix}$$

1. Since this subroutine assumes a constant drag parameter α, all derivatives of α are set = 0. If it were desired to provide for a variable α, the seventh row of the matrix would have to be filled in accordingly.
2. See Appendix A for relations used in calculation of derivatives of $\underline{A}$ matrix $\partial \underline{f}(\underline{x})/\partial \underline{x}$.

$$B = \omega^2 - W \qquad W = G_M/R^3 \quad (\text{where } R = [x^2 + y^2 + (z + R_E)^2]^{1/2})$$

$$C = 2\omega \sin(\text{DLAT})$$

$$E = -2\omega \cos(\text{DLAT})$$

$$G = \frac{C^2}{4} - W$$

$$J = \frac{(C)(E)}{4}$$

$$K = (J)(R_E)$$

$$N = \frac{E^2}{4} - W$$

$$Q = (N)(R_E)$$

The position and motion of the vehicle are described in terms of the state vector

$$\underline{x} = \begin{bmatrix} x \\ y \\ z \\ \dot{x} \\ \dot{y} \\ \dot{z} \\ \alpha \end{bmatrix}$$

The equations of motion are numerically integrated using a predictor-corrector method, which makes use of the state vector $\underline{x}$, the vector $\underline{b} = \underline{f}(\underline{x}) = \partial\underline{x}/\partial t$, and the matrix $\underline{A} = \partial\underline{f}(\underline{x})/\partial\underline{x}$. A block diagram of the integration scheme, proposed by M. Gruber,[2] is shown in Fig. 3. The vector $\underline{b}$ is written more explicitly below, while Table IV gives the matrix $\underline{A}$.

$$\underline{b} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \\ \ddot{x} \\ \ddot{y} \\ \ddot{z} \\ \dot{\alpha} \end{bmatrix}$$

Figure 3 shows that the vector $\underline{x}(t_k)$ at time t_k, is integrated over one step size to time t_{k+1} giving a predicted state vector $\underline{\xi}(t_{k+1})$ which is based on knowledge of the vectors $\underline{x}(t_k)$ and $\underline{x}(t_{k-1})$. A corrected vector $\underline{x}(t_{k+1})$ is then calculated, using the newly obtained prediction $\underline{\xi}(t_{k+1})$ and $\underline{x}(t_k)$. If the convergence errors, defined as $|\underline{\xi}(t_{k+1}) - \underline{x}(t_{k+1})|$, are less than the values specified in the ERRMAX array for all seven components of $\underline{x}$, $\underline{x}(t_{k+1})$ is accepted as

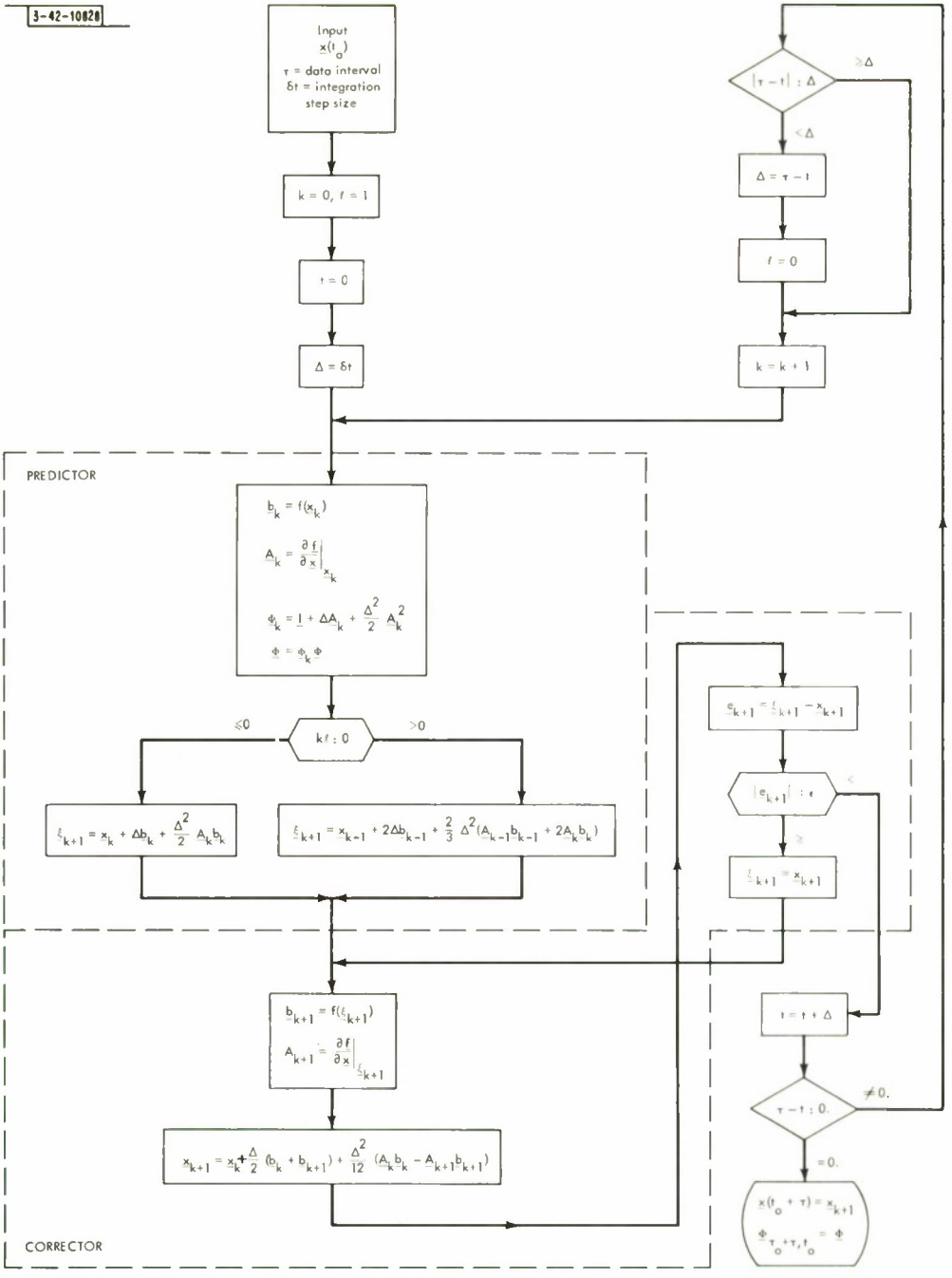

Fig. 3. Flow chart of TRAJGX calculations.

the value of $\underline{x}$ at time t_{k+1}. If not, $\underline{\xi}(t_{k+1})$ is set equal to $\underline{x}(t_{k+1})$ and another value of $\underline{x}(t_{k+1})$ is calculated based on $\underline{x}(t_k)$ and the corrected vector $\underline{\xi}(t_{k+1})$. New convergence errors are calculated and the comparison test and subsequent procedure are repeated until sufficiently small convergence errors are obtained. The above procedure is repeated for each integration step until the desired interval has been covered. Figure 4 is a flow chart of the subroutine.

V. SUBROUTINE ESTMAT

A. Description

Subroutine ESTMAT estimates the true value of a state vector by combining a predicted value, given in radar-centered rectangular coordinates, with a noisy measurement given in radar-centered polar coordinates. It is essentially a recursive Kalman-filter scheme. Double-precision calculations are used throughout. In making the estimate, the prediction and the measurement are weighted in proportion to the inverse of their respective mean-square error covariance matrices. ESTMAT may also be used to perform trajectory error analyses, in which the effects of hypothetical errors in the initial state and subsequent measurements are calculated. Suitable variances are inserted in the appropriate covariance matrices which are evaluated and propagated along the nominal trajectory.

The subroutine calculates the mean-square error covariance matrix of the estimate, and prints it out if desired. This matrix is saved by the subroutine to be used, along with the transition matrix, in computing the covariance matrix of the next prediction. The transition matrix, defined by the equation

$$\Phi_{k,\,k-1} = \partial\underline{x}(t_k)/\partial\underline{x}(t_{k-1})$$

is required as an input argument to ESTMAT. It is computed by subroutine TRAJGX in the course of calculating the predicted value of $\underline{x}(t_k)$. If TRAJGX is not used, a suitable transition matrix must be calculated elsewhere.

Subroutines DMTMUL, MINV, RCOVTX, XCOVTR, and XYTOR are also required by ESTMAT.

One problem that arises in linear recursive tracking is that at a sufficiently high data rate, the computed covariance matrix of the estimate will tend towards zero. This has the effect of giving progressively less weight to measurements. As a result of errors in our model of the state equation, or in our covariance matrix calculations, or in the numerical integration routine (especially during re-entry), it is often advisable to weight the measurements higher than the algorithm normally would.

An arbitrarily chosen device (others are possible) that has been used to remedy the situation is to prevent the determinant of the covariance matrix from decreasing below a given reference value. This reference is generally set equal to the determinant of the covariance matrix after a specified number of data points have been processed. For all subsequent estimates, the determinant of the covariance matrix is calculated and the entire covariance matrix is scaled up so that its determinant is equal to the reference. The determinant, and also the covariance matrix, are said to be "clamped," and prevented from decreasing to zero. A parameter KLAMP, referred to as the clamping time, or memory, of the algorithm specifies the number of data points to be processed before calculating the reference determinant.

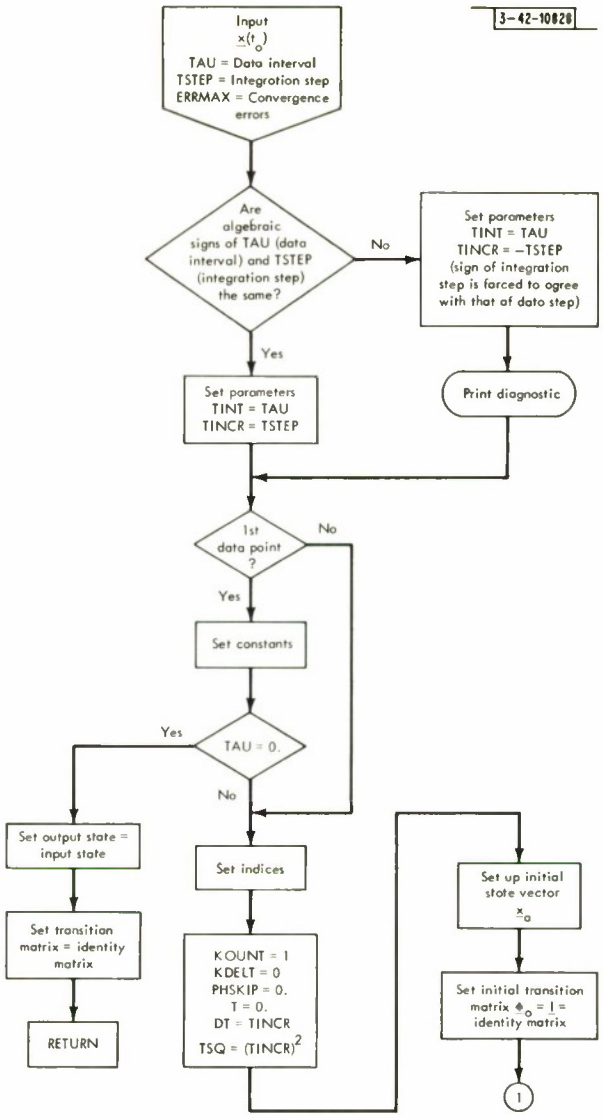

Fig. 4. Flow chart of subroutine TRAJGX.

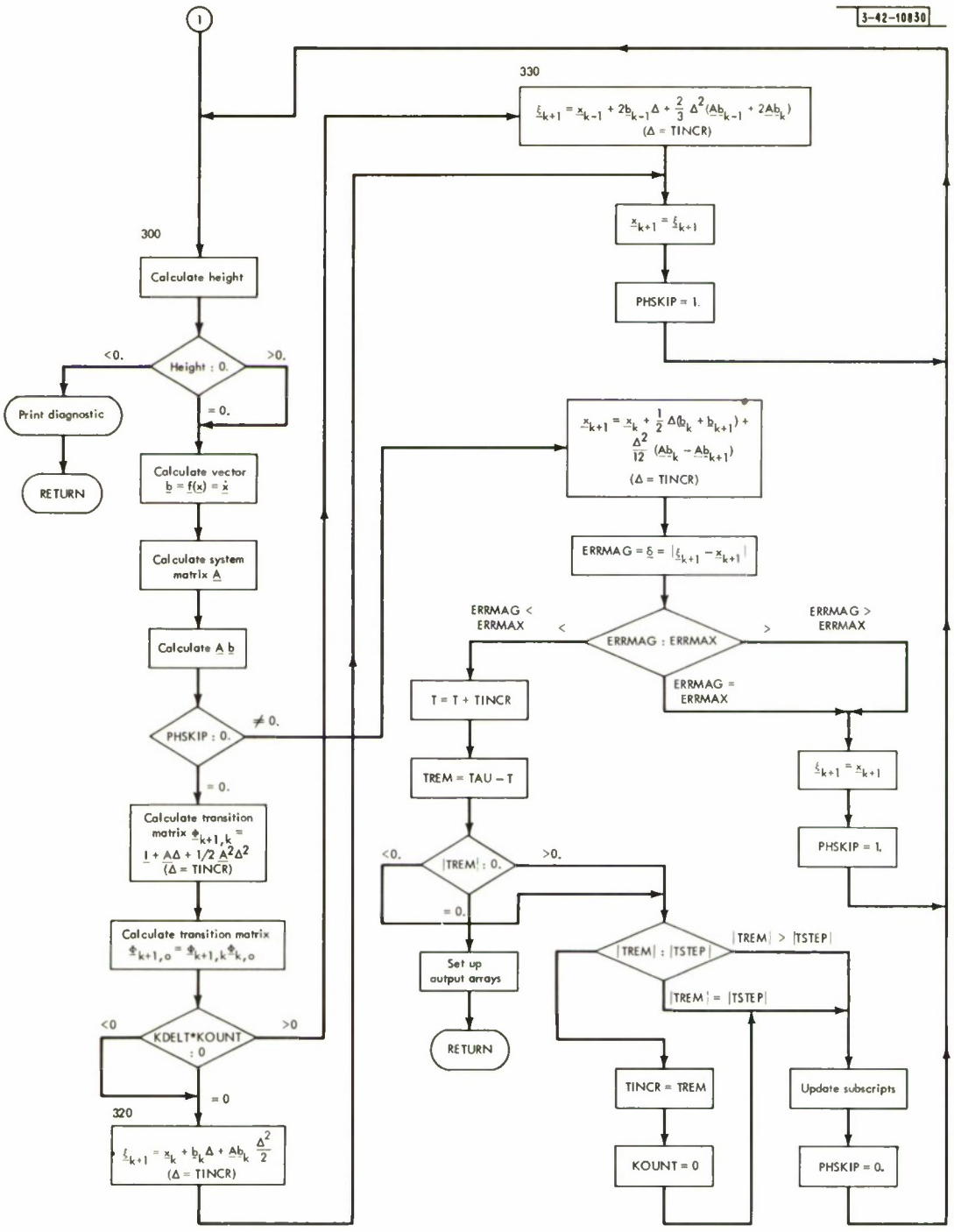

Fig. 4. Continued.

Subroutine ESTMAT is called by the following statement:

CALL ESTMAT (STATEM; STATEP, PHIMTX, ESTATE, DEVX, DEVR)

The input arguments are:

STATEM	Measurement vector in radar-centered polar coordinates STATEM(1) = range measurement STATEM(2) = azimuth measurement (in radians) STATEM(3) = elevation measurement (in radians) STATEM(4) = range rate (Doppler) measurement (if used) STATEM(5) = azimuth rate measurement (if used) STATEM(6) = elevation rate measurement (if used) STATEM(7) : generally not used
STATEP	Predicted state vector in radar-centered rectangular coordinates
PHIMTX	Transition matrix (Note: this matrix is destroyed by ESTMAT)

The output arguments are:

ESTATE	Estimated state vector in radar-centered rectangular coordinates
DEVX	Seven-component vector containing expected rms errors of estimated state vector in rectangular coordinates
DEVR	Seven-component vector containing expected rms errors of estimated state vector in radar polar coordinates

The following statements are also required in the calling program:

```
COMMON/ACOM/COVAR(7), SIGMA(7)/FCOM/COORD, DLAT, PRNT/
ICOM/KLAMP, MDATA, NN
DIMENSION STATEM(7), STATEP(7), ESTATE(7), PHIMTX(7, 7),
DEVX(7), DEVR(7)
```

The quantity MDATA is an integer specifying the mode of operation of ESTMAT, and may have values from −7 to +7, but not zero. (If MDATA = 0, noiseless trajectory data will be generated and subroutine ESTMAT will not be called.) The magnitude of MDATA indicates the number of measurement vector components used, while the sign indicates the type of job to be done – positive for estimation, negative for error analysis. Some common values are given below:

MDATA = 4	state-vector estimation, using range, azimuth, elevation, and range rate (Doppler) measurements
MDATA = 3	state-vector estimation, using range, azimuth, and elevation measurements
MDATA = 1	state-vector estimation, using range measurements only
MDATA = −4	error analysis, assuming range, azimuth, elevation, and range rate (Doppler) measurements
MDATA = −3	error analysis, assuming range, azimuth, and elevation measurements
MDATA = −1	error analysis, assuming range measurement only.

The SIGMA array contains the rms errors associated with the measurement vector.

SIGMA(1) = rms range measurement error

SIGMA(2) = rms azimuth measurement error (in radians)

SIGMA(3) = rms elevation measurement error (in radians)

SIGMA(4) = rms range rate (Doppler) measurement (if used)

SIGMA(5) = rms azimuth rate measurement error (if used)

SIGMA(6) = rms elevation rate measurement error (if used)

SIGMA(7) : not used

The COVAR array contains the rms errors associated with the initial estimate of the state vector. (At the start of a run, an initial estimate of the state vector, along with rms errors associated with it, must be given.) The elements of the COVAR array may be in radar-centered rectangular or polar coordinates and are listed below.

COVAR(1) = rms error in initial estimate of x (or range)

COVAR(2) = rms error in initial estimate of y (or azimuth)

COVAR(3) = rms error in initial estimate of z (or elevation)

COVAR(4) = rms error in initial estimate of $\dot{x}$ (or range rate)

COVAR(5) = rms error in initial estimate of $\dot{y}$ (or azimuth rate)

COVAR(6) = rms error in initial estimate of $\dot{z}$ (or elevation rate)

COVAR(7) = rms error in initial estimate of α = drag parameter = $1/\beta$

The specification of the coordinate system of COVAR is indicated by the value of the parameter COORD.

COORD = ±1.0 COVAR in rectangular coordinates

COORD = ±2.0 COVAR in polar coordinates

A selection of printed output is available by specification of the parameter PRNT.

PRNT < 0. No printed output from ESTMAT

PRNT = 0. The covariance matrix of the estimated state vector is printed for each point processed. It is given in rectangular coordinates along with a "hybrid" matrix in polar coordinates. In the hybrid matrix, the diagonal terms are rms, rather than mean-square, errors while off-diagonal terms are correlation coefficients of the mean-square errors. Mathematically,

$$e_{ii} = \sqrt{\sigma^2_i} \qquad \text{(diagonal terms)}$$

$$e_{ij} = \frac{\sigma^2_{ij}}{\sqrt{\sigma^2_{ii}\,\sigma^2_{jj}}} \qquad \text{(off-diagonal terms)}$$

where

σ^2 = mean square covariance in polar coordinates

e = hybrid matrix element

PRNT > 0. Covariance matrices for predicted as well as estimated state vectors are printed for each data point in the same formats described above. In addition, the partial derivative matrix H, the weighting matrix W (see Sec. V. B), and the estimated state vector in rectangular coordinates are printed.

Other quantities in COMMON storage are:

NN = number of the data point currently being processed.

KLAMP = clamping, or memory, time of process note: KLAMP = 0 is used to indicate that no clamping is desired. The same effect may be obtained by making KLAMP greater than the total number of points to be processed.

DLAT = latitude of radar (reference) site, in degrees (not used by ESTMAT)

B. Mathematical Discussion

As stated in part (A) of this section, ESTMAT estimates the true value of a state vector by combining a predicted value of the state vector with a noisy measurement vector. The mathematical formula is given below:

$$\hat{\underline{x}}_k = \underline{x}_{k,\,k-1} + \underline{W}\,[\underline{r}_k - \underline{h}(\underline{x}_{k,\,k-1})] \tag{1}$$

where

$$\underline{W} = (\underline{S}^{-1}_{k,\,k-1} + \underline{H}^T \underline{R}^{-1} \underline{H})^{-1}\, \underline{H}^T \underline{R}^{-1} \tag{2a}$$

$$= \underline{S}_k \underline{H}^T \underline{R}^{-1} \tag{2b}$$

$\hat{\underline{x}}_k$ = estimate of state vector $\underline{x}$ at time t_k

$\underline{x}_{k,\,k-1}$ = predicted state vector $\underline{x}$ at time t_k, based on $\hat{x}_{k-1}$, the estimate of $\underline{x}$ at time t_{k-1}

$\underline{r}_k$ = measurement vector at time t_k

$\underline{S}_k$ = mean-square error covariance matrix of $\hat{\underline{x}}_k$
where $(S_k)_{ij} = E\{[(\hat{x}_k)_i - (x^*_k)_i]\,[(\hat{x}_k)_j - (x^*_k)_j]^T\}$
and $\underline{x}^*$ = nominal (noiseless) value of $\underline{x}$.

$\underline{S}_{k,\,k-1}$ = mean-square error covariance matrix of $\underline{x}_{k,\,k-1}$

$\underline{R}$ = mean-square error covariance matrix of measurement vector $\underline{r}$.

$\underline{H}$ = partial derivative matrix $\partial \underline{r} / \partial \underline{x}$
where $\underline{r}$ = vector in measurement coordinates
$\underline{x}$ = vector in estimation coordinates
$\underline{r} = \underline{h}(\underline{x})$
If $\underline{r}$ has m components and $\underline{x}$ has n components $\underline{H}$ will have m rows and n columns. If $\underline{r}$ and $\underline{x}$ are in the same coordinate system, $H_{ij} = 1\delta(i - j)$

Equation (1) is approximately equivalent to

$$\hat{\underline{x}}_k \cong \underline{S}_k [\underline{S}^{-1}_{k,\,k-1}\, \underline{x}_{k,\,k-1} + \underline{H}^T \underline{R}^{-1} \underline{r}_k] \tag{3}$$

Equation (3) shows that the prediction $\underline{x}_{k,\,k-1}$ and the measurement $\underline{r}_k$ are each weighted by the inverse of their respective covariance function. The matrix $\underline{S}_k$ is related to the other matrices by the relation

$$\underline{S}_k = (\underline{S}_{k,\,k-1}^{-1} + \underline{H}^T \underline{R}^{-1} \underline{H})^{-1} \tag{4}$$

and $S_{k,\,k-1}$ is related to the covariance matrix of the estimate at the previous point by

$$\underline{S}_{k,\,k-1} = \underline{\Phi}_{k,\,k-1} \underline{S}_{k-1} \underline{\Phi}_{k,\,k-1}^T \tag{5}$$

where

$$\underline{\Phi}_{k,\,k-1} = \partial \underline{x}_{k,\,k-1} / \partial \hat{\underline{x}}_{k-1} \tag{6}$$

The matrix Φ is called the transition matrix and is calculated in subroutine TRAJGX.

Figure 5 shows a flow chart of version I of ESTMAT, based on Eqs. (1) and (2).

Emphasis is on the inverse covariance matrix S^{-1}, which is continually updated by the equation

$$\underline{S}_{k,\,k-1}^{-1} = \underline{\Phi}_{k,\,k-1}^{-1T} \underline{S}_{k-1}^{-1} \underline{\Phi}_{k,\,k-1}^{-1} \quad . \tag{7}$$

Matrix inversion is performed to obtain $\Phi_{k,\,k-1}^{-1}$ for use in Eq. (7) and to calculate the weighting matrix W of Eq. (2). Since $\underline{x}$ is a seven-component vector, both of these inversions involve 7×7 matrices.

V. O. Mowery[3] has presented some alternate equations, mathematically identical with Eqs. (2) and (4), which involve inversion of lesser order matrices. Replacing Eqs. (2) and (4), we have

$$\underline{W} = \underline{S}_{k,\,k-1} \underline{H}^T (\underline{R}_k + \underline{H} \underline{S}_{k,\,k-1} \underline{H}^T)^{-1} \tag{8}$$

$$\underline{S}_k = (\underline{I} - \underline{W}\underline{H})\, \underline{S}_{k,\,k-1} \tag{9}$$

where

$\underline{I}$ = identity matrix .

The matrix to be inverted is now $n \times n$ where n, the dimension of the measurement vector $\underline{r}$, is practically always less than 7. Version II of ESTMAT, using Mowery's equations, is shown in Fig. 6.

In performing error analyses, all matrices are evaluated along the nominal trajectory, with the covariance matrices being updated by Eqs. (7) and (4), or (5) and (9) depending on which version of ESTMAT was used.

VI. SUBROUTINE DENS

A. Description

Subroutine DENS computes, in double precision, the atmospheric density at any altitude. The subroutine may be entered through two points, DENS and ATM. The calling statements for each entry point are given below with explanation.

The statement CALL DENS must be used to enter the program initially, before any atmospheric density data are required. It sets up arrays of up to 100 discrete altitudes in km and kft, and arrays of their corresponding atmospheric densities in kg/meter^3 and lb/ft^3, respectively.*

* Atmospheric density data are from the U. S. Standard Atmosphere 1962 (United States Government Printing Office, Washington, D. C., 1962), Tables I and IV.

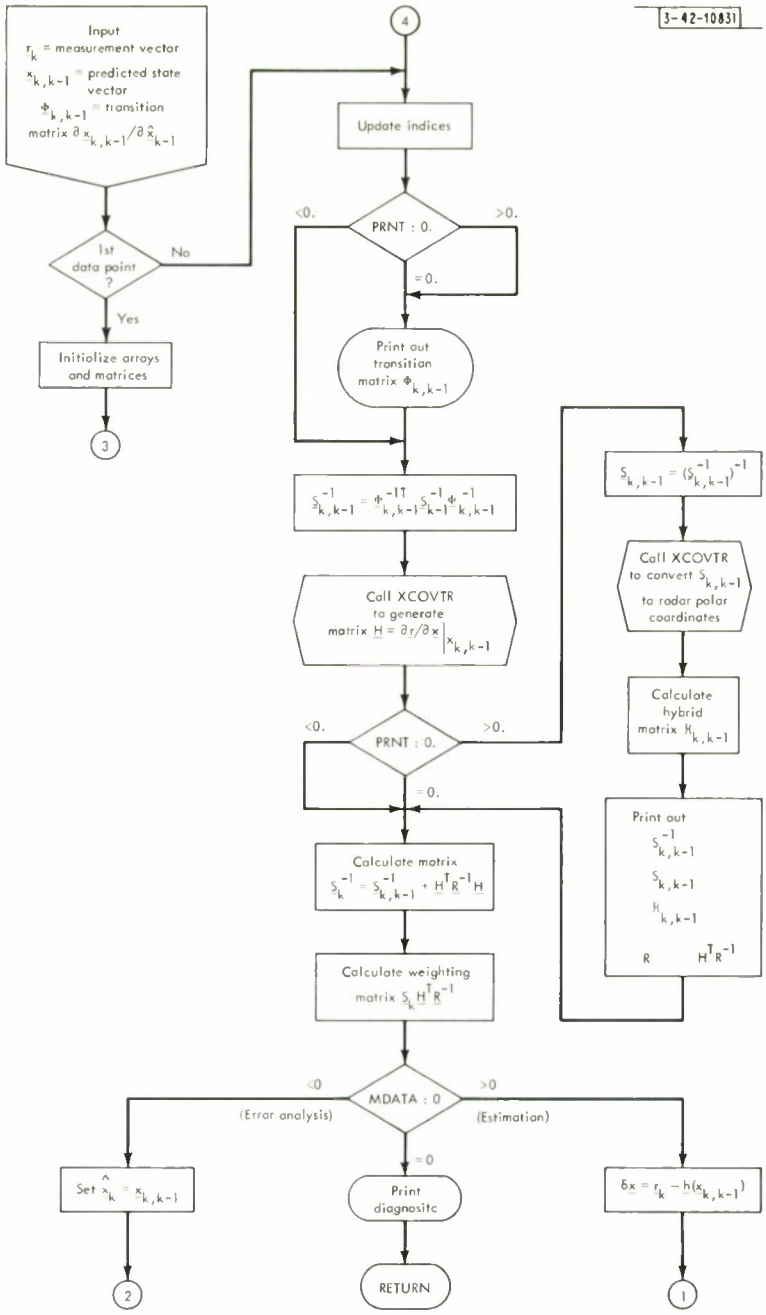

Fig. 5. Flow chart of subroutine ESTMAT.

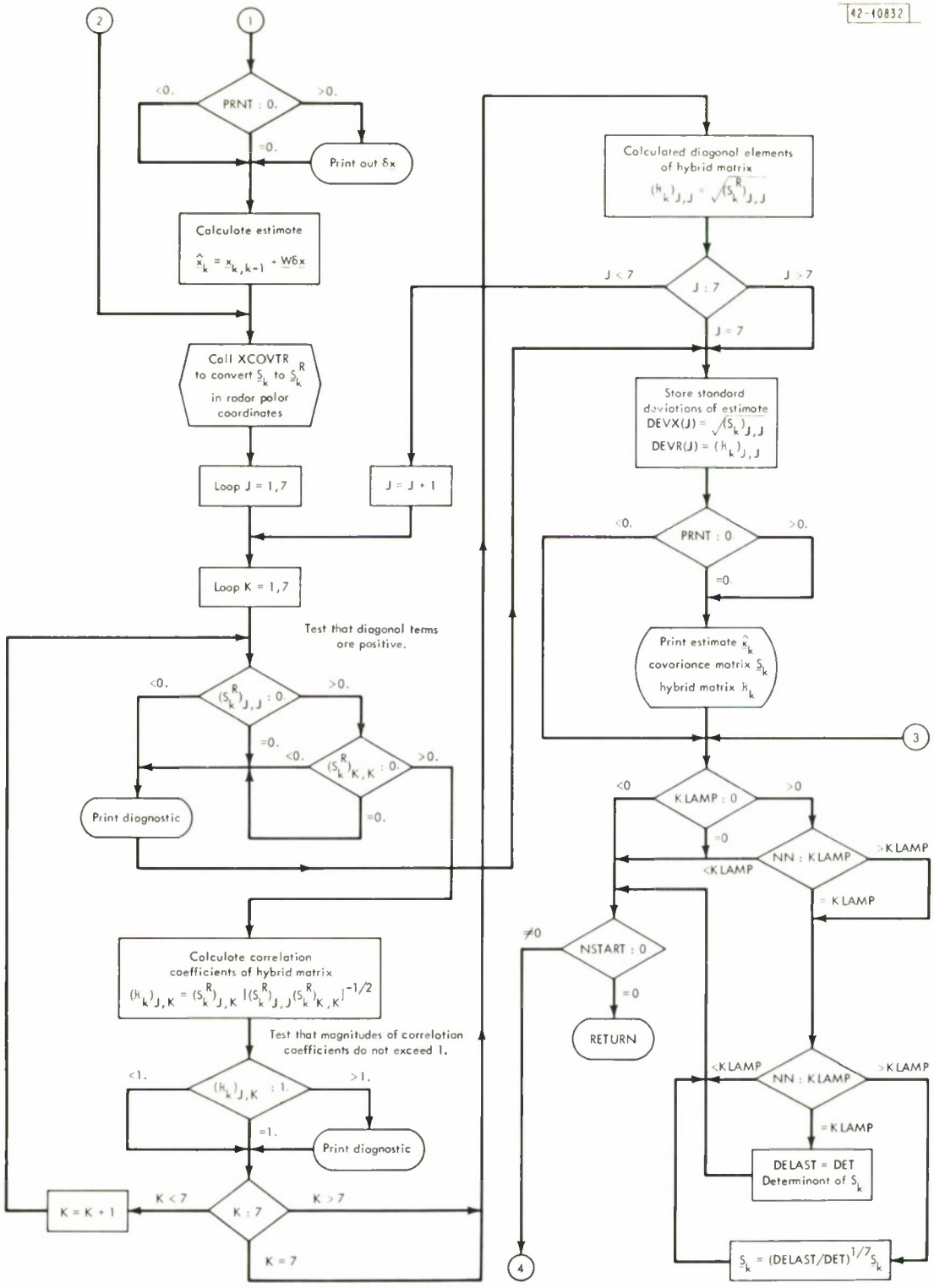

Fig. 5. Continued.

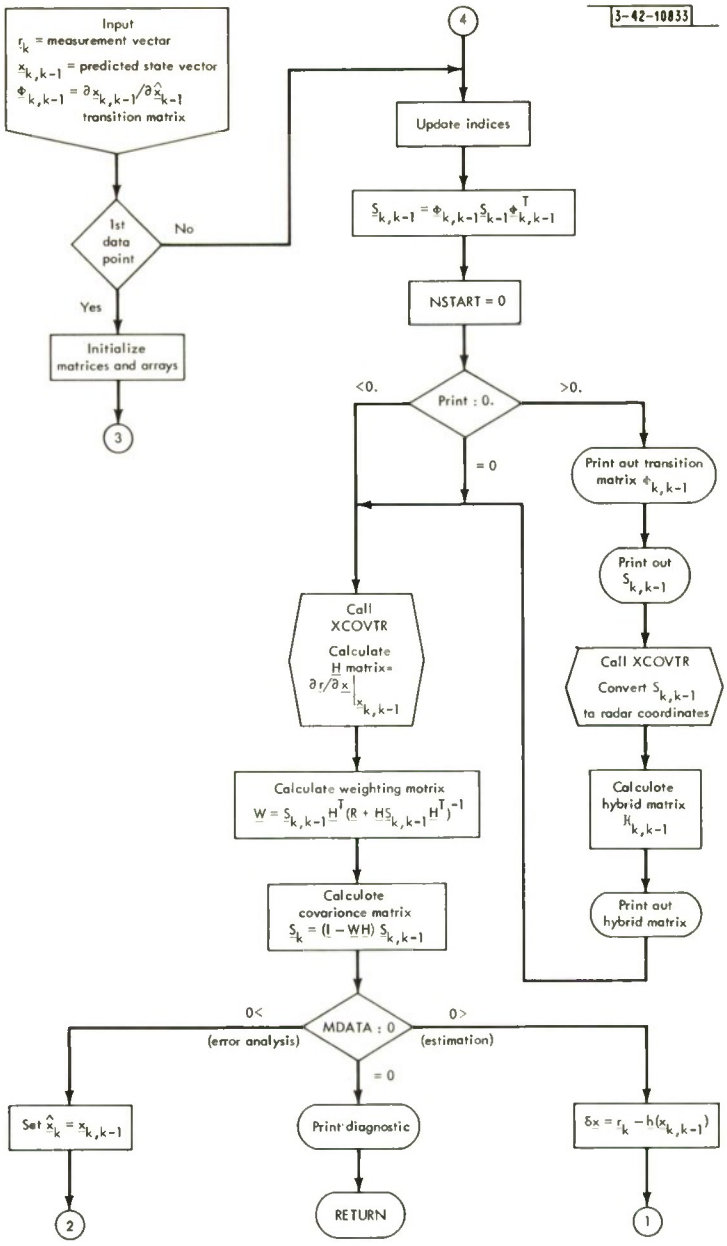

Fig. 6. Flow chart of subroutine ESTMAT (Mowery method).

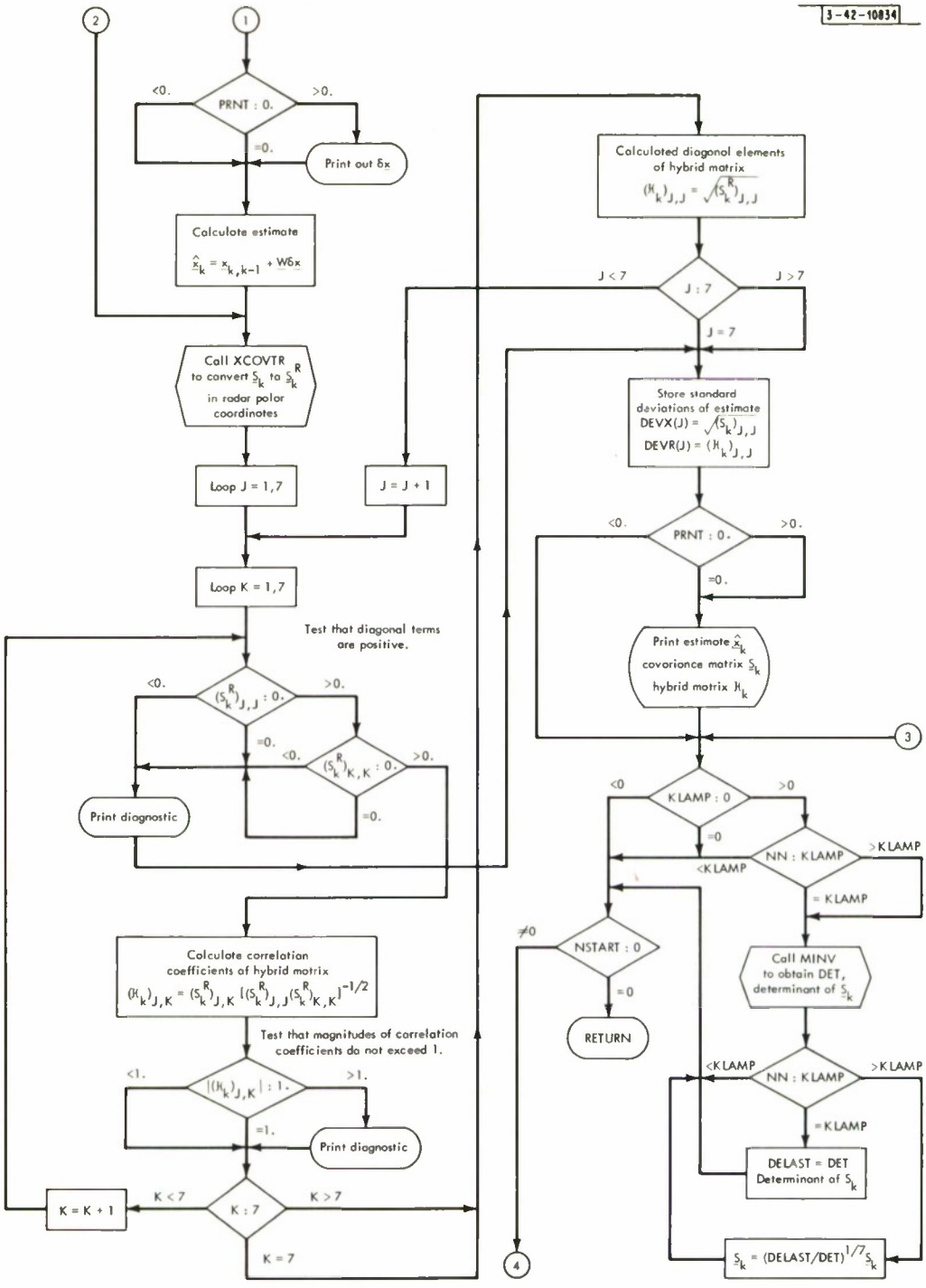

Fig. 6. Continued.

(As presently written, only the first 61 elements of each array are filled in, up to an altitude of about 200 km. At higher altitudes the atmospheric density is set equal to zero.) Control is then returned to the main program.

The statement CALL ATM(HKMFT, RHO) is used when atmospheric density data are required. The arguments are

HKMFT = altitude in km or kft

RHO = density in kg/m^3 or lb/ft^3

The units used will depend on the value of the parameter COORD, in COMMON storage. English units are used if COORD > 0, metric units if COORD < 0.

The following COMMON statement is also required in the calling program:

COMMON/FCOM/COORD, DLAT, PRNT

The other quantities in COMMON storage are not used by this subroutine.

B. Mathematical Discussion

The given altitude is compared with those stored by the subroutine, and the desired density is computed as follows:

If

$$h_j < H < h_{j+1}$$

then

$$A = (H - h_j)/(h_{j+1} - h_j)$$

and

$$\rho = \rho_j \left(\frac{\rho_{j+1}}{\rho_j}\right)^A$$

where

H = input altitude, in kilometers or kilofeet

h_j = j^{th} altitude in stored array

ρ_j = j^{th} atmospheric density in stored array, corresponding to h_j

ρ = atmospheric density at altitude H

If $H > h_{j\,max}$, where $h_{j\,max}$ = maximum altitude stored in array, then $\rho = 0$. If $H < 0$, the program prints out EARTH IMPACT.

VII. SUBROUTINE XYTOR

Subroutine XYTOR converts position and velocity given in radar-centered rectangular coordinates into radar-centered polar coordinates. The following calling statement is required:

CALL XYTOR(X, Y, Z, XDOT, YDOT, ZDOT, R, A, E, RDOT, ADOT, EDOT)

The input arguments are:

X = position component in easterly direction

Y = position component in northerly direction

Z = position component in vertical direction

XDOT = $\dot{X}$ = dX/dt

YDOT = $\dot{Y}$ = dY/dt

ZDOT = $\dot{Z}$ = dZ/dt

The output arguments are:

R = range

A = azimuth (in radians)

E = elevation (in radians)

RDOT = range rate = $\dot{R}$ = dR/dt

ADOT = azimuth rate (radians/sec) = $\dot{A}$ = dA/dt

EDOT = elevation rate (radians/sec) = $\dot{E}$ = dE/dt

The following equations are used in the calculations:

$$R = (X^2 + Y^2 + Z^2)^{1/2}$$

$$A = \tan^{-1}(X/Y)$$

$$E = \sin^{-1}(Z/R)$$

$$RDOT = (X\dot{X} + Y\dot{Y} + Z\dot{Z})/R$$

$$ADOT = (Y\dot{X} + X\dot{Y})/(X^2 + Y^2)$$

$$EDOT = (R\dot{Z} - Z\dot{R})/[R(R^2 - Z^2)^{1/2}]$$

VIII. SUBROUTINE RTOXY

Subroutine RTOXY converts position and velocity given in radar-centered polar coordinates to radar-centered rectangular coordinates. The following calling statement is required:

CALL RTOXY(R, A, E, RDOT, ADOT, EDOT, X, Y, Z, XDOT, YDOT, ZDOT)

The input arguments are:

R = range

A = azimuth (in radians)

E = elevation (in radians)

RDOT = range rate = $\dot{R}$ = dR/dt

ADOT = azimuth rate (radians/sec) = $\dot{A}$ = dA/dt

EDOT = elevation rate (radians/sec) = $\dot{E}$ = dE/dt

The output arguments are:

X = position component in easterly direction

Y = position component in northerly direction

Z = position component in vertical direction

XDOT = $\dot{X}$ = dX/dt

YDOT = $\dot{Y}$ = dY/dt

ZDOT = $\dot{Z}$ = dZ/dt

The following equations are used:

$X = R \cos E \sin A$

$Y = R \cos E \cos A$

$Z = R \sin E$

$XDOT = \dot{X}R/R - Z\dot{E} \sin A + Y\dot{A}$

$YDOT = \dot{Y}R/R - Z\dot{E} \cos A - X\dot{A}$

$ZDOT = \dot{Z}R/R + R\dot{E} \cos E$

IX. SUBROUTINES XCOVTR AND RCOVTX

A. Description of XCOVTR

Subroutine XCOVTR converts a 7 × 7 mean-square error covariance matrix in radar-centered rectangular coordinates to one in polar coordinates. It also calculates a partial derivative matrix $\underline{H} = \partial \underline{r}/\partial \underline{x}$ where $\underline{r}$ and $\underline{x}$ are the state vectors in radar-centered polar and rectangular coordinates, respectively.

The calling statement is

CALL XCOVTR(XCOV, XVCTR, RCOV, NCODE)

The input arguments are:

XCOV = 7 × 7 covariance matrix in rectangular coordinates

XVCTR = seven-component state vector used in evaluating matrices

NCODE = operation selector code

NCODE > 0: subroutine calculates covariance matrix in polar coordinates

NCODE < 0: subroutine calculates partial derivative matrix $\underline{H} = \partial \underline{r}/\partial \underline{x}$

The output argument is:

RCOV = 7 × 7 mean-square error covariance in radar-centered polar coordinates

or

partial derivative matrix $H = \partial \underline{r}/\partial \underline{x}$.

The following statement is also required in the calling program:

```
DIMENSION XCOV(7, 7), RCOV(7, 7), XVCTR(7)
```

Subroutines XYTOR and DMTMUL are required by this subroutine.

B. Description of RCOVTX

Subroutine RCOVTX converts a 7×7 mean-square error covariance matrix in radar-centered polar coordinates to one in rectangular coordinates. The calling statement is:

```
CALL RCOVTX(COVR, XSTATE, COVX)
```

The input arguments are:

COVR = 7×7 mean-square error covariance matrix in polar coordinates

XSTATE = seven-component state vector used in evaluating matrices

The output argument is:

COVX = 7×7 mean-square error covariance matrix in radar-centered rectangular coordinates

The following statement is also required in the calling program:

```
DIMENSION COVR(7, 7), COVX(7, 7), XSTATE(7)
```

Subroutine XYTOR is required by this subroutine.

C. Mathematical Discussion

Consider the state vectors $\underline{x}$ and $\underline{r}$ in radar-centered rectangular and polar coordinates, respectively.

$\underline{x}$:	$\underline{r}$:
x_1 = x easterly component	$r_1 = R$ = range
x_2 = y northerly component	$r_2 = A$ = azimuth
x_3 = z vertical component	$r_3 = E$ = elevation
$x_4 = \dot{x}$	$r_4 = \dot{R}$
$x_5 = \dot{y}$	$r_5 = \dot{A}$
$x_6 = \dot{z}$	$r_6 = \dot{E}$
$x_7 = \alpha = 1/\beta$ = reciprocal of ballistic coefficient	$r_7 = \alpha = 1/\beta$

The covariance matrices are

$$\underline{\Sigma}_x = E\left[(\underline{x} - \underline{x}^*)(\underline{x} - \underline{x}^*)^T\right] \text{ in rectangular coordinates}$$

$$\underline{\Sigma}_r = E\left[(\underline{r} - \underline{r}^*)(\underline{r} - \underline{r}^*)^T\right] \text{ in polar coordinates}$$

where

E indicates average or expected value

$\underline{x}^*$ = nominal value of $\underline{x}$

$\underline{r}^*$ = nominal value of $\underline{r}$

The covariance matrices $\underline{\Sigma}_x$ and $\underline{\Sigma}_r$ are related by the equations

$$\underline{\Sigma}_r = \underline{H}\underline{\Sigma}_x\underline{H}^T$$

$$\underline{\Sigma}_x = \underline{B}\underline{\Sigma}_r\underline{B}^T$$

$$\underline{H} = \partial\underline{r}/\partial\underline{x}$$

$$H_{ij} = \partial r_i/\partial x_i$$

$$\underline{B} = \partial\underline{x}/\partial\underline{r} = H^{-1}$$

$$B_{ij} = \partial x_i/\partial r_j$$

where the subscripts indicate the component (or element) of the vector (or matrix) involved.

The first six components of the $\underline{x}$ and $\underline{r}$ vectors are related by the equations given with subroutines XYTOR and RTOXY, while $x_7 = r_7$. The various elements of the H and B matrices are given below:

$$H_{11} = \frac{\partial R}{\partial x} = x/R$$

$$H_{12} = \frac{\partial R}{\partial y} = y/R$$

$$H_{13} = \frac{\partial R}{\partial z} = z/R$$

$$H_{14} = \text{through } H_{17} = 0$$

$$H_{21} = \frac{\partial A}{\partial x} = \frac{y}{x^2 + y^2}$$

$$H_{22} = \frac{\partial A}{\partial y} = -\frac{x}{x^2 + y^2}$$

$$H_{23} = \frac{\partial A}{\partial z} = 0$$

$$H_{24} \text{ through } H_{27} = 0$$

$$H_{31} = \frac{\partial E}{\partial x} = -\frac{xz}{R^2\sqrt{x^2 + y^2}}$$

$$H_{32} = \frac{\partial E}{\partial y} = -\frac{yz}{R^2\sqrt{x^2 + y^2}}$$

$$H_{33} = \frac{\partial E}{\partial z} = \frac{\sqrt{x^2 + y^2}}{R^2}$$

$$H_{34} \text{ through } H_{37} = 0$$

$$H_{41} = \frac{\partial \dot{R}}{\partial x} = \frac{R\dot{x} - \dot{R}x}{R^2}$$

$$H_{42} = \frac{\partial \dot{R}}{\partial y} = \frac{R\dot{y} - \dot{R}y}{R^2}$$

$$H_{43} = \frac{\partial \dot{R}}{\partial z} = \frac{R\dot{z} - \dot{R}z}{R^2}$$

$$H_{44} = H_{11}$$

$$H_{45} = H_{12}$$

$$H_{46} = H_{13}$$

$$H_{47} = 0$$

$$H_{51} = \frac{\partial \dot{A}}{\partial x} = -\frac{(2x\dot{A} + \dot{y})}{x^2 + y^2}$$

$$H_{52} = \frac{\partial \dot{A}}{\partial y} = \frac{\dot{x} - 2\dot{A}y}{x^2 + y^2}$$

$$H_{53} = \frac{\partial \dot{A}}{\partial z} = 0$$

$$H_{54} = H_{21}$$

$$H_{55} = H_{22}$$

$$H_{56} = H_{23}$$

$$H_{57} = 0$$

$$H_{61} = \frac{\partial \dot{E}}{\partial x} = -\frac{x\dot{E}}{(x^2 + y^2)} - \frac{2\dot{R}}{R} H_{31} - \frac{z\dot{x}}{R^2\sqrt{x^2 + y^2}}$$

$$H_{62} = \frac{\partial \dot{E}}{\partial y} = -\frac{y\dot{E}}{(x^2 + y^2)} - \frac{2\dot{R}}{R} H_{32} - \frac{z\dot{y}}{R^2\sqrt{x^2 + y^2}}$$

$$H_{63} = \frac{\partial \dot{E}}{\partial z} = -\frac{z\dot{E}}{R^2} - \frac{\dot{R}}{R} H_{33}$$

$$H_{64} = H_{31}$$

$$H_{65} = H_{32}$$

$$H_{66} = H_{33}$$

$$H_{67} = 0$$

H_{71} through $H_{76} = 0$

$$H_{77} = \partial\alpha/\partial\alpha = 1$$

$$B_{11} = \partial x/\partial R = \cos E \sin A$$

$$B_{12} = \partial x/\partial A = R \cos E \cos A = y$$

$$B_{13} = \partial x/\partial E = -R \sin E \sin A = -z \sin A$$

B_{14} through $B_{17} = 0$

$$B_{21} = \partial y/\partial R = \cos E \cos A$$

$$B_{22} = \partial y/\partial A = -R \cos E \sin A = -x$$

$$B_{23} = \partial y/\partial E = -R \sin E \cos A = z \cos A$$

B_{24} through $B_{27} = 0$

$$B_{31} = \partial z/\partial R = \sin E = z/R$$

$$B_{32} = \partial z/\partial A = 0$$

$$B_{33} = \partial z/\partial E = R \cos E$$

B_{34} through $B_{37} = 0$

$$B_{41} = \partial \dot{x}/\partial R = \dot{A} \cos E \cos A - \dot{E} \sin E \sin A$$

$$B_{42} = \partial \dot{x}/\partial A = \dot{R} \cos E \cos A - \dot{E}R \sin E \cos A - \dot{A}R \cos E \sin A$$

$$= \dot{R} \cos E \cos A - z\dot{E} \cos A - x\dot{A}$$

$$B_{43} = -\dot{R} \sin E \sin A - \dot{E}R \cos E \sin A - \dot{A}R \sin E \cos A$$

$$= -\dot{R} \sin E \sin A - x\dot{E} - z\dot{A} \cos A$$

$$B_{44} = \partial\dot{x}/\partial\dot{R} = \partial x/\partial R = B_{11}$$

$$B_{45} = \partial\dot{x}/\partial\dot{A} = \partial x/\partial A = B_{12}$$

$$B_{46} = \partial\dot{x}/\partial\dot{E} = \partial x/\partial E = B_{13}$$

$$B_{47} = 0$$

$$B_{51} = \partial\dot{y}/\partial R = -\dot{E} \sin E \cos A - \dot{A} \cos E \sin A$$

$$B_{52} = \partial\dot{y}/\partial A = -\dot{R} \cos E \sin A + \dot{E}R \sin E \sin A - AR \cos E \cos A$$

$$= -\dot{R} \cos E \sin A + z\dot{E} \sin A - y\dot{A}$$

$$B_{53} = \partial\dot{y}/\partial E = -\dot{R} \sin E \cos A - \dot{E}R \cos E \cos A + \dot{A}R \sin E \sin A$$

$$= -\dot{R} \sin E \cos A - y\dot{E} + z\dot{A} \sin A$$

$$B_{54} = \partial\dot{y}/\partial\dot{R} = \partial y/\partial R = B_{21}$$

$$B_{55} = \partial\dot{y}/\partial\dot{A} = \partial y/\partial A = B_{22}$$

$$B_{56} = \partial\dot{y}/\partial\dot{E} = \partial y/\partial E = B_{23}$$

$$B_{57} = 0$$

$$B_{61} = \partial\dot{z}/\partial R = \dot{E} \cos E$$

$$B_{62} = \partial\dot{z}/\partial A = 0$$

$$B_{63} = \partial\dot{z}/\partial E = \dot{R} \cos E - \dot{E}R \sin E$$

$$= \dot{R} \cos E - z\dot{E}$$

$$B_{64} = \partial\dot{z}/\partial\dot{R} = \partial z/\partial R = B_{31}$$

$$B_{65} = \partial\dot{z}/\partial\dot{A} = \partial z/\partial A = B_{32}$$

$$B_{66} = \partial\dot{z}/\partial\dot{E} = \partial z/\partial E = B_{33}$$

$$B_{67} = 0$$

$$B_{71} \text{ through } B_{76} = 0$$

$$B_{77} = \partial\alpha/\partial\alpha = 1 \quad .$$

X. SUBROUTINE GAUSSN

A. Description

Subroutine GAUSSN generates random numbers with a Gaussian distribution of zero mean and standard deviation SIGMA. The program is called by

CALL GAUSSN(NS, RN, SIGMA)

where

NS = number of samples desired

RN = array name of output

SIGMA = desired standard deviation

B. Mathematical Discussion

The subroutine makes use of the library subroutines RAN and ALOG. The procedure used to obtain a Gaussian number, RN, is as follows:[4]

Let $y = -\ln V_1$

$z = -\ln V_2$

where V_1 and V_2 are random numbers obtained from the RAN subroutine.

The quantity $(y-1)^2$ is compared with 2z. If $(y-1)^2 > 2z$, two new random numbers are selected and the above comparison repeated. If $(y-1)^2 \leq 2z$, a third random number S is selected to determine the sign of the desired output number. If

$S < 0.5$, RN = (y) (SIGMA)

$S > 0.5$, RN = − (y) (SIGMA)

S = 0.5, another number S is chosen and the above test for sign is repeated.

XI. SUBROUTINE DMTMUL

Subroutine DMTMUL is a double-precision matrix multiplication subroutine. It is called by

CALL DMTMUL(A, B, AB, NRA, NCA, NCB).

The input arguments are:

A, B } input matrices to be multiplied

NRA = number of rows in matrix A

NCA = number of columns in matrix A

NCB = number of columns in matrix B .

The output argument is

AB = matrix product (A)(B) .

The matrices A, B, and AB must have the same dimensions in both the calling program and the subroutine. When used with the TRAP program described here, A, B, and AB are dimensioned 7×7. Smaller matrices may also be multiplied by DMTMUL, even though they may not fill the complete array, by proper specification of NRA, NCA, and NCB.

XII. SUBROUTINE MINV

Subroutine MINV is a matrix inversion subroutine from the IBM System/360 Scientific Subroutine Package.[5] It uses the standard Gauss-Jordan method and is available in both single and

double precision. (Our applications use the double-precision version.) The subroutine is called by the statement:

```
CALL MINV(A, N, D, L, M)
```

where

A = input matrix, destroyed in computation and replaced by resultant inverse

N = order of matrix

D = resultant determinant

L, M = work vectors of length N .

The one-dimensional array variables A, L and M must be suitably dimensioned in the calling program.

The matrix to be inverted may be stored in a one-dimensional array of size N^2 before calling MINV.[6] A series of FORTRAN statements to do this are given below:

```
      DO 1  J = 1, N
      DO 1  K = 1, N
      L = N*(J - 1) + K
    1 A(L) = ARRAY(J, K)
```

where ARRAY is the two-dimensional array of the matrix to be inverted.

Subroutine MINV may now be called to invert the matrix stored in the one-dimensional array A. After matrix inversion, the resulting inverse, stored in the one-dimensional array A, may be placed in a two-dimensional array by the following statements:

```
      DO 2  L = 1, LMAX
      J = (L - 1)/N + 1
      K = L - N*(J - 1)
    2 ARRAYI(J, K) = A(L)
```

where LMAX = N^2 and ARRAYI is the $N \times N$ array of the inverted matrix.

ACKNOWLEDGMENT

The author wishes to thank Mr. M. Gruber, Dr. J. A. Tabaczynski, and Dr. R. P. Wishner for helpful discussions and suggestions.

REFERENCES

1. S. F. Catalano and H. Schneider, "Determination of Weight-to-Drag Ratio from Radar Measurements," Group Report 47G-6, Lincoln Laboratory, M.I.T., (8 March 1963), p. 13.
2. M. Gruber, "An Approach to Target Tracking," TN 1967-8, Lincoln Laboratory, M.I.T. (10 February 1967).
3. V. O. Mowery, "Least Squares Recursive Differential Correction Estimation in Nonlinear Problems," IEEE Trans. Automat. Control AC-10, 399 (1965).
4. H. Kahn, "Application of Monte Carlo," RM-1237-AFC, Rand Corporation, pp. 39 – 40 (19 April 1954, revised 27 April 1956).
5. System/360 Scientific Subroutine Package (360 A-CM-03X) Version II Programmers' Manual, No. H20-0205-2 (IBM, 1967).
6. Ref. 5, pp. 3 – 4.

APPENDIX A
SOME RELATIONS USED IN CALCULATION OF DERIVATIVES FOR $\underline{A}$ MATRIX $\partial \underline{f}(\underline{x})/\partial \underline{x}$

$$P = -\frac{\rho V_D}{2}$$

$\rho = \rho_o e^{-\gamma h}$ = atmospheric density

$h = [x^2 + y^2 + (z + R_E)^2]^{1/2} - R_E$ = altitude

R_E = radius of earth

$V_D = (\dot{x}^2 + \dot{y}^2 + \dot{z}^2)^{1/2}$ = drag velocity

$R = [x^2 + y^2 + (z + R_E)^2]^{1/2}$ = distance from center of earth to object

$$\partial P/\partial x = -P\gamma \frac{x}{R}$$

$$\partial P/\partial y = -P\gamma \frac{y}{R}$$

$$\partial P/\partial z = -P\gamma(z + R_E)/R$$

$$\partial P/\partial \dot{x} = P\dot{x}/V_D^2$$

$$\partial P/\partial \dot{y} = P\dot{y}/V_D^2$$

$$\partial P/\partial \dot{z} = P\dot{z}/V_D^2$$

$$B = \omega^2 - G_M/R^3$$

$$G = \omega^2 \sin^2(DLAT) - G_M/R^3$$

$$N = \omega^2 \cos^2(DLAT) - G_M/R^3$$

$$\partial B/\partial x = \partial N/\partial x = 3xG_M/R^5$$

$$\partial B/\partial y = \partial G/\partial y = \partial N/\partial y = 3yG_M/R^5$$

$$\partial B/\partial z = \partial g/\partial z = \partial N/\partial z = 3(z + R_E)\, G_M/R^5$$

APPENDIX B

PROGRAM LISTINGS

TABLE B-I

MAIN PROGRAM (VERSION I)

```
C      MAIN PROGRAM
       IMPLICIT REAL*8(A-H,O-Z)
       COMMON /ACOM/COVAR(7),SIGMA(7)/FCOM/COORD,DLAT,PRNT/ICOM/KLAMP,MDA
      1TA,NN
       DIMENSION GNOISE(7),RGN(54),AZN(54),ELN(54),RAPN(54),BETAH(54),
      1      RHAT(54),AHAT(54),EHAT(54),RPHAT(54),APHAT(54),EPHAT(54),STAT
      2EM(7),STATEP(7),PHIMAT(7,7),FNOISE(7),ALPHA(54),DALPH(54)
       DIMENSION PHINV(7,7),PROD(7,7),ESTATE(7),STATEI(7),X(54),Y(54),Z(5
      14),XP(54),YP(54),ZP(54),DX(54),DY(54),DZ(54),DXP(54),DYP(54),DZP(5
      24),XHAT(54),YHAT(54),ZHAT(54),XPHAT(54),YPHAT(54),ZPHAT(54),ASPECT
      3(54),SIGR(54),SIGA(54),SIGE(54),SIGRP(54),SIGAP(54),SIGEP(54),SIGX
      4(54),SIGY(54),SIGZ(54),SIGXP(54),SIGYP(54),SIGZP(54),SIGALP(54)
       DIMENSION GN(7),LABEL(18),RA(54),AZ(54),EL(54),      AZP(54),ELP(5
      14),HEIGHT(54),TIME(54),RAP(54),BATA(54),ERRMAX(7) ,STDEVR(7),STDEV
      2X(7)
       DIMENSION DR(54),DE(54),DA(54),DRP(54),DAP(54),DEP(54),DB(54)
C
C      CALL DENS TO SET UP ATMOSPHERIC DENSITY TABLES
C
       CALL DENS
C
C      READ INPUT DATA CARDS
C
    1  READ(5,2,END=9000)(LABEL(I),I=1,18)
    2  FORMAT (18A4)
       READ(5,3)(STATEI(J),J=1,7)
    3  FORMAT(7F10.3)
       READ(5,4)TZERO,DELT,TINT,TINCR,DLAT,PRNT,NDATA,KLAMP,MDATA
    4  FORMAT(5F10.3,F5.3,3I5)
       READ(5,3)(SIGMA(J),J=1,6),COORD
        READ(5,3)(COVAR(J),J=1,7)
       READ(5,3)(GNOISE(J),J=1,7)
       READ(5,3)(ERRMAX(JJ),JJ=1,7)
C
C        LABEL= IDENTIFYING COMMENTS TO APPEAR WITH DATA PRINT-OUT
C        TZERO= INITIAL TIME
C         TINT=INTERVAL BETWEEN DESIRED DATA POINTS (SEC.)
C         TINCR=INTEGRATION STEP SIZE (SEC.)
C         DLAT= LATITUDE OF RADAR SITE (IN DEGREES)
C         NDATA=NUMBER OF DESIRED DATA POINTS
C        PRNT=PRINT-OUT SELECTOR
C           PRNT=-1. MINIMUM PRINT-OUT, TABULATED SUMMARY ONLY
C           PRNT= 0. PRINT-OUT OF COVARIANCE MATRICES + TABULATED SUMMARY
C           PRNT= 1. FULL PRINT-OUT
C        KLAMP=MEMORY OF ALGORITHM,EXPRESSED AS NUMBER OF DATA POINTS
C        MDATA=MODE SELECTOR
C           MDATA GREATER THAN 0.  TRAJECTORY PARAMETER ESTIMATION
C           MDATA            = 0.  NOISELESS TRAJECTORY GENERATION ONLY
C           MDATA LESS THAN = 0.  ERROR ANALYSIS
C        COORD = COORDINATE AND UNIT SELECTOR
C           COORD=-2. POLAR COORDINATES, METRIC UNITS
C           COORD=-1. RECTANGULAR COORDINATES, METRIC UNITS
C           COORD=+1. RECTANGULAR COORDINATES, ENGLISH UNITS
C           COORD=+2. POLAR COORDINATES, ENGLISH UNITS
C        SIGMA = RMS NOISE COMPONENTS OF MEASUREMENT VECTOR
C        COVAR = RMS NOISE COMPONENTS OF INITIAL STATE VECTOR
```

TABLE B-I (Continued)

```
C        FNOISE = NOISE SAMPLES ASSOCIATED WITH INITIAL STATE VECTOR
C        ERRMAX = TRAJECTORY INTEGRATION CONVERGENCE ERRORS (IN REC-
C           TANGULAR COORDINATES)
C
    9  IMAX=50
       COORDM=DABS(COORD)
       RTDEG=180./3.14159265
C
C      SET UP INITIAL STATE IN BOTH RADAR POLAR AND XYZ COORDINATES
C
       IF(COORDM-1.)400,400,300
  300  RA(1)=STATEI(1)
       AZ(1)=STATEI(2)
       EL(1)=STATEI(3)
       RAP(1)=STATEI(4)
       AZP(1)=STATEI(5)
       ELP(1)=STATEI(6)
       CALL RTOXY(RA(1),AZ(1),EL(1),RAP(1),AZP(1),ELP(1),X(1),Y(1),Z(1),X
      1P(1),YP(1),ZP(1))
       GO TO 401
  400  X(1)= STATEI(1)
       Y(1)= STATEI(2)
       Z(1)= STATEI(3)
       XP(1)=STATEI(4)
       YP(1)=STATEI(5)
       ZP(1)=STATEI(6)
       CALL XYTOR(X(1),Y(1),Z(1),XP(1),YP(1),ZP(1),RA(1),AZ(1),EL(1),RAP(
      11),AZP(1),ELP(1))
  401  BETA=1./STATEI(7)
       BATA(1)=BETA
       BETA1=BATA(1)
C
C      CALCULATE ASPECT ANGLE
C
       ASP=(X(1   )*XP(1   )+Y(1   )*YP(1   )+Z(1   )*ZP(1   ))/(RA(1   )*DSQRT(
      1XP(1   )**2+YP(1   )**2+ZP(1   )**2))
       ASPECT(1   )=180.-RTDEG*DARCOS(ASP)
       TIME(1)=TZERO
       TLAST=TZERO
C
C      PRINT HEADER PAGE OF OUTPUT LISTING
C
       WRITE (6,10)(LABEL(I),I=1,18)
   10  FORMAT('1',20X,'OUTPUT LISTING'/18A4//)
       WRITE(6,11)
   11  FORMAT(/5X,'INITIAL CONDITIONS'//)
       IF(COORD)14,16,16
   14  RE=2.0925738D7/3.2808333
       WRITE(6,15)
   15  FORMAT(/5X,'METRIC UNITS(METERS,KILOGRAMS,RADIANS,SECONDS) USED TH
      1ROUGHOUT PROGRAM'//)
       GO TO 18
   16  RE=2.0925738D7
       WRITE(6,17)
   17  FORMAT(/5X,'ENGLISH UNITS(FEET,POUNDS,RADIANS,SECONDS) USED THROUG
      1HOUT PROGRAM'//)
   18  DO 19 J=1,7
   19  STATEM(J)=0.
       X1=X(1)
       Y1=Y(1)
```

TABLE B-I (Continued)

```
      Z1=Z(1)
      XDOT1=XP(1)
      YDOT1=YP(1)
      ZDOT1=ZP(1)
      HEIGHT(1)=DSQRT(X1**2+Y1**2+(Z1+RE)**2)-RE
      INDEX=1
      WRITE(6,50)TIME(INDEX),RA(INDEX),AZ(INDEX),EL(INDEX),RAP(INDEX),AZ
     1P(INDEX),ELP(INDEX),X1,Y1,Z1,XDOT1,YDOT1,ZDOT1,HEIGHT(INDEX),BETA1
      WRITE(6,500)ASPECT(1)
  500 FORMAT(5X,'ASPECT ANGLE =',F10.2,' DEGREES'//)
      WRITE(6,110)DLAT,NDATA,TINT,TINCR
  110 FORMAT(      /5X,'RADAR LATITUDE',3X,F10.5,2X,'DEGREES'/5X,I6,5X,'
     1DATA POINTS SPACED',3X, F10.6 ,2X,'SEC. APART, INTEGRATION STEP',2
     2X,F10.6,2X,'SEC.'//)
      IF(MDATA)70,74,72
   70 WRITE(6,71)
   71 FORMAT(/5X,'ERROR ANALYSIS'//)
  198 DO 201 J=1,7
  201 ENOISE(J)=0.
      GO TO 75
   74 WRITE(6,80)
   80 FORMAT(/5X,'TRAJECTORY GENERATION'//)
      TAU=TINT
      GO TO 81
   72 WRITE(6,73)
   73 FORMAT(/5X,'TRAJECTORY PARAMETER ESTIMATION FROM SIMULATED NOISY D
     1ATA'//)
   75 IF(KLAMP)760,760,751
  751 WRITE(6,752) KLAMP
  752 FORMAT(/5X,'MEMORY TIME =',I6,' DATA POINTS'//)
      TSTEP=TINCR
  760 WRITE(6,76) DELT
   76 FORMAT(/5X,'TIME OF INITIAL MEASUREMENT WITH RESPECT TO TIME OF IN
     1ITIAL ESTIMATE      =',F10.3,3X,'SEC.'//)
      WRITE(6,77)
   77 FORMAT(/5X,'RMS NOISE LEVELS ASSOCIATED WITH MEASUREMENT VECTOR'/)
      WRITE (6,7)(SIGMA(J),J=1,6)
    7 FORMAT(/5X,'SIGMA(R)=',F10.2,5X,'SIGMA(AZ)=',F10.6,5X,'SIGMA(EL)='
     1,F10.6,5X,'SIGMA(RDOT)=',F10.2/5X,'SIGMA(AZDOT)=',F10.6,5X,'SIGMA(
     2ELDOT)=',F10.6/)
      WRITE(6,13) MDATA
   13 FORMAT(/5X,'ONLY THE 1ST',I3,1X,'QUANTITIES OF THE VECTOR'/
     15X,'CONSISTING OF R,AZ,EL,RDOT,AZDOT,ELDOT,1./BETA, ARE MEASURED')
      IF(COORDM-1.)780,780,782
  780 WRITE(6,781)
  781 FORMAT(/5X,'INITIAL STATE VECTOR READ IN RADAR-CENTERED RECTANGULA
     1R COORDINATES'//)
      GO TO 784
  782 WRITE(6,783)
  783 FORMAT(/5X,'INITIAL STATE VECTOR READ IN RADAR-CENTERED POLAR COOR
     1DINATES'//)
  784 WRITE(6,78)(COVAR(J),J=1,7)
   78 FORMAT(/5X,'RMS NOISE LEVELS ASSOCIATED WITH INITIAL STATE VECTOR'
     1/7D17.8//)
      WRITE(6,799)(ENOISE(J),J=1,7)
  799 FORMAT(/5X,'NOISE SAMPLES AT INITIAL STATE'/7D17.8//)
   81 WRITE(6,12)(ERRMAX(JJ),JJ=1,7)
   12 FORMAT(/5X,'MAXIMUM INTEGRATION CONVERGENCE ERRORS(IN RECTANGULAR
     1COORDINATES'/7D17.8///)
      L=2
```

TABLE B-I (Continued)

```
C
C       DATA LOOP
C
        DO 1000 NN=1,NDATA
        NN=NN
        IF(NN-1)21,20,21
   20   IF(NDATA)202,22,202
C
C       ADD NOISE OF RMS LEVEL COVAR TO NOMINAL INITIAL STATE
C        (NOISE=0. FOR ERROR ANALYSIS CASE)
C       SET INITIAL STATE ESTIMATE = INITIAL NOISY STATE
C
  202   IF(COORDM-1.)203,203,204
  203   XHAT(1) =X(1)+FNOISE(1)
        YHAT(1) =Y(1)+FNOISE(2)
        ZHAT(1) =Z(1)+FNOISE(3)
        XPHAT(1)=XP(1)+FNOISE(4)
        YPHAT(1)=YP(1)+FNOISE(5)
        ZPHAT(1)=ZP(1)+FNOISE(6)
        CALL XYTOR(XHAT(1),YHAT(1),ZHAT(1),XPHAT(1),YPHAT(1),ZPHAT(1),RHAT
       1(1),AHAT(1),EHAT(1),RPHAT(1),APHAT(1),EPHAT(1))
        GO TO 206
  204   RHAT(1)=RA(1)+FNOISE(1)
        AHAT(1)=AZ(1)+FNOISE(2)
        EHAT(1)=EL(1)+FNOISE(3)
        RPHAT(1)=RAP(1)+FNOISE(4)
        APHAT(1)=AZP(1)+FNOISE(5)
        EPHAT(1)=ELP(1)+FNOISE(6)
        CALL RTOXY(RHAT(1),AHAT(1),EHAT(1),RPHAT(1),APHAT(1),EPHAT(1),XHAT
       1(1),YHAT(1),ZHAT(1),XPHAT(1),YPHAT(1),ZPHAT(1))
  206   ESTATE(7)=STATEI(7)+FNOISE(7)
        ESTATE(1)=XHAT(1)
        ESTATE(2)=YHAT(1)
        ESTATE(3)=ZHAT(1)
        ESTATE(4)=XPHAT(1)
        ESTATE(5)=YPHAT(1)
        ESTATE(6)=ZPHAT(1)
        ALPHA(1)=ESTATE(7)
  205   BETAH(1)=1./ESTATE(7)
        TAU=DELT
C
C       COMPUTE INITIAL ERRORS = (ESTIMATE)-(NOMINAL STATE)
C
        DR(1)=RHAT(1)-RA(1)
        DA(1)=AHAT(1)-AZ(1)
        DE(1)=EHAT(1)-EL(1)
        DRP(1)=RPHAT(1)-RAP(1)
        DAP(1)=APHAT(1)-AZP(1)
        DEP(1)=EPHAT(1)-ELP(1)
        DB(1)=BETAH(1)-BATA(1)
        DALPH(1)=ESTATE(7)-1./BETA
        DX(1)=XHAT(1)-X(1)
        DY(1)=YHAT(1)-Y(1)
        DZ(1)=ZHAT(1)-Z(1)
        DXP(1)=XPHAT(1)-XP(1)
        DYP(1)=YPHAT(1)-YP(1)
        DZP(1)=ZPHAT(1)-ZP(1)
        GO TO 22
C
C       SET UP ARGUMENTS FOR TRAJGX TO INTEGRATE NOMINAL TRAJECTORY TO
```

TABLE B-I (Continued)

```
C      NEXT POINT
C
   21 X1=X2
      Y1=Y2
      Z1=Z2
      XDOT1=XDOT2
      YDOT1=YDOT2
      ZDOT1=ZDOT2
      BETA1=BETA2
      TAU=TINT
C
C      UPDATE SUBSCRIPTS
C
   22 K=MOD(NN,IMAX)
      NXT=K+1
      INDEX=NXT
      IF(K)23,23,24
   23 K=IMAX
C
C      CALL TRAJGX TO GENERATE NEXT DATA POINT OF NOMINAL TRAJECTORY
C       IN XYZ COORDINATES
C
   24 TIME(NXT)=TLAST+TAU
      TLAST=TIME(INDEX)
      CALL TRAJGX(X1,Y1,Z1,XDOT1,YDOT1,ZDOT1,BETA1,TAU ,TINCR,ERRMAX,
     1X2,Y2,Z2,XDOT2,YDOT2,ZDOT2,BETA2,PHIMAT)
      X(NXT)=X2
      Y(NXT)=Y2
      Z(NXT)=Z2
      XP(NXT)=XDOT2
      YP(NXT)=YDOT2
      ZP(NXT)=ZDOT2
      BATA(NXT)=BETA2
      HEIGHT(NXT)=DSQRT(X2**2+Y2**2+(Z2+RE)**2)-RE
C
C      CONVERT TO RADAR POLAR COORDINATES
C
      CALL XYTOR(X2,Y2,Z2,XDOT2,YDOT2,ZDOT2,RA(NXT),AZ(NXT),EL(NXT),RAP(
     1NXT),AZP(NXT),ELP(NXT))
C
C      CALCULATE ASPECT ANGLE
C
      ASP=(X(NXT)*XP(NXT)+Y(NXT)*YP(NXT)+Z(NXT)*ZP(NXT))/(RA(NXT)*DSQRT(
     1XP(NXT)**2+YP(NXT)**2+ZP(NXT)**2))
      ASPECT(NXT)=180.-RTDEG*DARCOS(ASP)
C
C       WRITE NEXT NOMINAL DATA POINT
C
      IF(PRNT)502,48,48
   48 WRITE(6,49)
   49 FORMAT('1',5X,'NOMINAL DATA POINT'//)
      WRITE(6,50)TIME(INDEX),RA(INDEX),AZ(INDEX),EL(INDEX),RAP(INDEX),AZ
     1P(INDEX),ELP(INDEX),X2,Y2,Z2,XDOT2,YDOT2,ZDOT2,HEIGHT(INDEX),BETA2
   50 FORMAT(5X,'TIME =',F10.4/5X,'RA =',F15.2,3X,'AZ =',F10.6,3X,'EL ='
     1,F10.6,3X,'RAP=',F10.2,3X,'AZP=',F10.6,3X,'ELP=',F10.6/5X,'X  =',F
     210.2,3X,'Y  =',F10.2,3X,'Z  =',F10.2,3X,'XP =',F10.2,3X,'YP =',F10
     3.2,3X,'ZP =',F10.2/5X,'HEIGHT =',5X,F15.2,20X,'BETA=',5X,F10.2//)
      WRITE(6,500)ASPECT(NXT)
C
C       WRITE TRANSITION MATRIX FOR NOMINAL DATA POINT
```

TABLE B-I (Continued)

```
C
  501 WRITE(6,51)((PHIMAT(JJ,KK),KK=1,7),JJ=1,7)
   51 FORMAT(5X,'TRANSITION MATRIX PHI'/(7D17.8))
  502 IF(MDATA)5800,25,2400
C
C     ERROR ANALYSIS
C
 5800 STATEP(1)=X2
      STATEP(2)=Y2
      STATEP(3)=Z2
      STATEP(4)=XDOT2
      STATEP(5)=YDOT2
      STATEP(6)=ZDOT2
      STATEP(7)=1./BETA2
      CALL ESTMAT(STATEP,STATEP,PHIMAT,ESTATE,STDEVX,STDEVR)
C
C     STORE EXPECTED RMS ERRORS
C
      SIGR (NXT)=STDEVR(1)
      SIGA (NXT)=STDEVR(2)
      SIGE (NXT)=STDEVR(3)
      SIGRP(NXT)=STDEVR(4)
      SIGAP(NXT)=STDEVR(5)
      SIGEP(NXT)=STDEVR(6)
      SIGX (NXT)=STDEVX(1)
      SIGY (NXT)=STDEVX(2)
      SIGZ (NXT)=STDEVX(3)
      SIGXP(NXT)=STDEVX(4)
      SIGYP(NXT)=STDEVX(5)
      SIGZP(NXT)=STDEVX(6)
      SIGALP(NXT)=STDEVX(7)
      GO TO 25
C
C     TRACKING AND ESTIMATION.
C
C     ADD NOISE OF RMS LEVEL SIGMA TO OBTAIN SIMULATED MEASUREMENT
C
 2400 DO 241 J=1,4
  241 CALL GAUSSN(1,GNOISE(J),SIGMA(J))
      RGN(NXT)=RA(NXT)+GNOISE(1)
      AZN(NXT)=AZ(NXT)+GNOISE(2)
      ELN(NXT)=EL(NXT)+GNOISE(3)
      RAPN(NXT)=RAP(NXT)+GNOISE(4)
      STATEM(1)=RGN(NXT)
      STATEM(2)=AZN(NXT)
      STATEM(3)=ELN(NXT)
      STATEM(4)=RAPN(NXT)
      IF(PRNT)59,57,57
C
C     PRINT OUT SIMULATED NOISY MEASUREMENT
C
   57 WRITE(6,58) TIME(NXT),RGN(NXT),AZN(NXT),ELN(NXT),RAPN(NXT)
   58 FORMAT(/5X,'NOISY DATA'/5X,'TIME =',F10.4,5X,'RGN=',F10.2,3X,'AZN=
     1',F10.6,3X,'ELN=',F10.6,3X,'RAPN=',F10.2//)
C
C      CALL TRAJGX TO OBTAIN NEXT PREDICTED DATA POINT
C
   59 CALL TRAJGX(ESTATE(1),ESTATE(2),ESTATE(3),ESTATE(4),ESTATE(5),ESTA
     1TE(6),DELTAT(K),TAU,TINCR,ERRMAX,STATEP(1),STATEP(2),STATEP(3),STAT
     2EP(4),STATEP(5),STATEP(6),PBETA,PROD)
```

TABLE B-I (Continued)

```
C
C      CONVERT NEXT PREDICTED DATA POINT TO RADAR POLAR COORDINATES
C
       STATEP(7)=1./PBETA
       IF(PRNT)5702,5700,5700
 5700 CALL XYTCR(STATEP(1),STATEP(2),STATEP(3),STATEP(4),STATEP(5),STATE
     1P(6),PRA,PAZ,PEL,PRAP,PAZP,PELP)
       PH=DSQRT(STATEP(1)**2+STATEP(2)**2+(STATEP(3)+RE)**2)-RE
       WRITE(6,570)
  570 FORMAT(/5X,'PREDICTED STATE      BASED ON PAST DATA ONLY')
       WRITE(6,50)TIME(NXT),PRA,PAZ,PEL,PRAP,PAZP,PELP,STATEP(1),STATEP(2
     1),STATEP(3),STATEP(4),STATEP(5),STATEP(6),PH,PBETA
C
C      CALL ESTMAT TO COMBINE SIMULATED MEASUREMENT WITH PREDICTED POINT
C       TO OBTAIN FINAL ESTIMATE OF NEXT DATA POINT
C
 5702 CALL ESTMAT(STATEM,STATEP,PROD,ESTATE,STDEVX,STDEVR)
C
C      STORE EXPECTED RMS ERRORS
C
       SIGR (NXT)=STDEVR(1)
       SIGA (NXT)=STDEVR(2)
       SIGE (NXT)=STDEVR(3)
       SIGRP(NXT)=STDEVR(4)
       SIGAP(NXT)=STDEVR(5)
       SIGEP(NXT)=STDEVR(6)
       SIGX (NXT)=STDEVX(1)
       SIGY (NXT)=STDEVX(2)
       SIGZ (NXT)=STDEVX(3)
       SIGXP(NXT)=STDEVX(4)
       SIGYP(NXT)=STDEVX(5)
       SIGZP(NXT)=STDEVX(6)
       SIGALP(NXT)=STDEVX(7)
  579 CONTINUE
C
C      CONVERT ESTIMATED STATE TO RADAR POLAR COORDINATES
C      STORE ESTIMATED STATE VECTOR IN OUTPUT ARRAYS
C
 5793 CALL XYTCR(ESTATE(1),ESTATE(2),ESTATE(3),ESTATE(4),ESTATE(5),ESTAT
     1E(6),RHAT(NXT),AHAT(NXT),EHAT(NXT),RPHAT(NXT),APHAT(NXT),EPHAT(NXT
     2))
       XHAT(NXT)=ESTATE(1)
       YHAT(NXT)=ESTATE(2)
       ZHAT(NXT)=ESTATE(3)
       XPHAT(NXT)=ESTATE(4)
       YPHAT(NXT)=ESTATE(5)
       ZPHAT(NXT)=ESTATE(6)
       ALPHA(NXT)=ESTATE(7)
       BETAH(NXT)=1./ESTATE(7)
       HNXT=DSQRT(ESTATE(1)**2+ESTATE(2)**2+(ESTATE(3)+RE)**2)-RE
       IF(PRNT)581,5794,5794
C
C      PRINT OUT ESTIMATED STATE
C
 5794 WRITE(6,580)
  580 FORMAT(/5X,'ESTIMATED POINT'/)
       WRITE(6,50)TIME(NXT),RHAT(NXT),AHAT(NXT),EHAT(NXT),RPHAT(NXT),APHA
     1T(NXT),EPHAT(NXT),ESTATE(1),ESTATE(2),ESTATE(3),ESTATE(4),ESTATE(5
     2),ESTATE(6),HNXT,BETAH(NXT)
C
```

TABLE B-I (Continued)

```
C       COMPUTE DIFFERENCES BETWEEN ESTIMATE AND   NOMINAL DATA POINT
C
  581 DR(NXT)=RHAT(NXT)-RA(NXT)
      DA(NXT)=AHAT(NXT)-AZ(NXT)
      DE(NXT)=EHAT(NXT)-EL(NXT)
      DRP(NXT)=RPHAT(NXT)-RAP(NXT)
      DAP(NXT)=APHAT(NXT)-AZP(NXT)
      DEP(NXT)=EPHAT(NXT)-ELP(NXT)
      DB(NXT)=BETAH(NXT)-BATA(NXT)
      DALPH(NXT)=ESTATE(7)-1./BATA(NXT)
      DX(NXT)=XHAT(NXT)-X(NXT)
      DY(NXT)=YHAT(NXT)-Y(NXT)
      DZ(NXT)=ZHAT(NXT)-Z(NXT)
      DXP(NXT)=XPHAT(NXT)-XP(NXT)
      DYP(NXT)=YPHAT(NXT)-YP(NXT)
      DZP(NXT)=ZPHAT(NXT)-ZP(NXT)
C
C       TEST FOR LAST DATA POINT OF RUN
C
   25 IF(NN-NDATA)27,26,26
   26 KMAX=INDEX
      GO TO 40
   27 IF(INDEX-IMAX)1000,28,28
   28 KMAX=IMAX
C
C       WRITE OUTPUT
C
   40 WRITE(6,10)(LABEL(I),I=1,18)
      WRITE(6,41)
   41 FORMAT(/5X,'NOMINAL TRAJECTORY IN RADAR COORDINATES'//
     1           3X,'TIME',8X,'RANGE',7X,'RADOT',7X,'AZIM',8X,'AZDOT',7X
     2,'ELEV',8X, 'ELDOT',7X,'HEIGHT',6X,'BETA'//)
      WRITE(6,42)(TIME(KK),RA(KK),RAP(KK),AZ(KK),AZP(KK),EL(KK),ELP(KK),
     1HEIGHT(KK),BATA(KK),KK=1,KMAX)
   42 FORMAT(                                        3X,F10.3,2X,F10.2,X,F10
     2.2,2X,F10.6,2X,F10.6,2X,F10.6,2X,F10.6,2X,F10.2,2X,F10.2)
      WRITE(6,10)(LABEL(I),I=1,18)
      WRITE(6,410)
  410 FORMAT(5X,'NOMINAL TRAJECTORY IN XYZ COORDINATES'//)
      WRITE(6,411)
  411 FORMAT(3X,'TIME',8X,'X',11X,'Y',11X,'Z',11X,'XDOT',8X,'YDOT',8X,'Z
     1DOT',8X,'BETA',8X,'ASPECT ANGLE'//)
      WRITE(6,412)(TIME(J),X(J),Y(J),Z(J),XP(J),YP(J),ZP(J),BATA(J),ASPE
     1CT(J),J=1,KMAX)
  412 FORMAT(3X,F10.3,2X,F10.2,2X,F10.2,2X,F10.2,2X,F10.2,2X,F10.2,2X,F1
     10.2,2X,F10.2,2X,F10.2)
      IF(NDATA)900,1000,420
  420 WRITE(6,10)(LABEL(I),I=1,18)
      WRITE(6,43)
   43 FORMAT(5X,'MEASUREMENT DATA'//6X,'TIME',8X,'RANGE',7X,'RADOT',17X,
     1'AZIM',20X,'ELEV'//)
      WRITE(6,44)(TIME(KK),RGN(KK),RAPN(KK),AZN(KK),ELN(KK),KK=1,KMAX)
   44 FORMAT(3X,F10.3,2X,F10.2,2X,F10.2,14X,F10.6,14X,F10.6)
      WRITE (6,10)(LABEL(I),I=1,18)
      WRITE(6,60)
   60 FORMAT(5X,'ESTIMATED VALUES'//)
      WRITE(6,61)
   61 FORMAT(3X,'TIME',8X,'RANGE',7X,'RADOT',7X,'AZIM',8X,'AZDOT',7X,'EL
     1EV',8X,'ELDOT',7X,'BETA',8X,'ALPHA'//)
      WRITE(6,62)(TIME(J),RHAT(J),RPHAT(J),AHAT(J),APHAT(J),EHAT(J),EPHA
```

TABLE B-I (Continued)

```
     1T(J),8ETAH(J),ALPHA(J),J=1,KMAX)
  62 FORMAT(3X,F10.3,2X,F10.2,2X,F10.2,2X,F10.6,2X,F10.6,2X,F10.6,2X,F1
     10.6,2X,F10.2,2X,F12.8)
      WRITE(6,10)(LABEL(I),I=1,18)
      WRITE(6,60)
      WRITE(6,413)
 413 FORMAT(3X,'TIME',8X,'X',11X,'Y',11X,'Z',11X,'XDOT',8X,'YDOT',8X,'Z
     1DOT',8X,'BETA',8X,'ALPHA'//)
      WRITE(6,414)(TIME(J),XHAT(J),YHAT(J),ZHAT(J),XPHAT(J),YPHAT(J),ZPH
     1AT(J),BETAH(J),ALPHA(J),J=1,KMAX)
 414 FORMAT(3X,F10.3,2X,F10.2,2X,F10.2,2X,F10.2,2X,F10.2,2X,F10.2,2X,F1
     10.2,2X,F10.2,2X,F12.8)
      WRITE (6,10)(LABEL(I),I=1,18)
      WRITE(6,63)
  63 FORMAT(5X,'(ERRORS)=(ESTIMATED VALUES)-(NOMINAL VALUES)'//)
      WRITE(6,61)
      WRITE(6,62)(TIME(J),DR(J),DRP(J),DA(J),DAP(J),DE(J),DEP(J),DB(J),D
     1ALPH(J),J=1,KMAX)
      WRITE(6,10)(LABEL(I),I=1,18)
      WRITE(6,63)
      WRITE(6,413)
      WRITE(6,414)(TIME(J),DX(J),DY(J),DZ(J),DXP(J),DYP(J),DZP(J),DB(J),
     1DALPH(J),J=1,KMAX)
900   WRITE(6,10)(LABEL(I),I=1,18)
      WRITE(6,901)
901   FORMAT(5X,'EXPECTED RMS ERRORS'//)
      WRITE(6,61)
      WRITE(6,902)(TIME(J),SIGR(J),SIGRP(J),SIGA(J),SIGAP(J),SIGE(J),SIG
     1EP(J),SIGALP(J),J=L,KMAX)
902   FORMAT(3X,F10.3,2X,F10.3,2X,F10.2,2X,F10.6,2X,F10.6,2X,F10.6,2X,F1
     10.6,14X,F12.8)
      WRITE(6,10)(LABEL(I),I=1,18)
      WRITE(6,901)
      WRITE(6,413)
      WRITE(6,903)(TIME(J),SIGX(J),SIGY(J),SIGZ(J),SIGXP(J),SIGYP(J),SIG
     1ZP(J),SIGALP(J),J=L,KMAX)
903   FORMAT(3X,F10.3,2X,F10.2,2X,F10.2,2X,F10.2,2X,F10.2,2X,F10.2,2X,F1
     10.2,14X,F12.8)
      L=1
1000  CONTINUE
1001  GO TO 1
9000  RETURN
      END
```

TABLE B-II

MAIN PROGRAM (VERSION II)

```
C     MAIN PROGRAM
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON /ACOM/COVAR(7),SIGMA(7)/FCOM/COORD,DLAT,PRNT/ICOM/KLAMP,MDA
     1TA,NN
      DIMENSION GNOISE(7),RGN(54),AZN(54),ELN(54),RAPN(54),BETAH(54),
     1     RHAT(54),AHAT(54),EHAT(54),RPHAT(54),APHAT(54),EPHAT(54),STAT
     2EM(7),STATEP(7),PHIMAT(7,7),FNOISE(7),ALPHA(54),DALPH(54)
      DIMENSION PHINV(7,7),PROD(7,7),ESTATE(7),STATEI(7),X(54),Y(54),Z(5
     14),XP(54),YP(54),ZP(54),DX(54),DY(54),DZ(54),DXP(54),DYP(54),DZP(5
     24),XHAT(54),YHAT(54),ZHAT(54),XPHAT(54),YPHAT(54),ZPHAT(54),ASPECT
     3(54),SIGR(54),SIGA(54),SIGE(54),SIGRP(54),SIGAP(54),SIGEP(54),SIGX
     4(54),SIGY(54),SIGZ(54),SIGXP(54),SIGYP(54),SIGZP(54),SIGALP(54)
      DIMENSION GN(7),LABEL(18),RA(54),AZ(54),EL(54),      AZP(54),ELP(5
     14),HEIGHT(54),TIME(54),RAP(54),BATA(54),ERRMAX(7),STDEVR(7),STDEVX
     1(7)
      DIMENSION DR(54),DE(54),DA(54),DRP(54),DAP(54),DEP(54),DB(54)
C
C     CALL DENS TO SET UP ATMOSPHERIC DENSITY TABLES
C
      CALL DENS
C
C     READ INPUT PARAMETERS AND PRINT ON LISTING
C
    1 READ(5,2,END=9000)(LABEL(I),I=1,18)
    2 FORMAT (18A4)
      READ(5,4)TZERO,TINT,TINCR,DLAT,PRNT,NDATA,KLAMP,MDATA
    4 FORMAT(4F10.3,F5.3,3I5)
      READ(5,3)(SIGMA(J),J=1,6),COORD
    3 FORMAT(7F10.3)
      READ(5,3)(ERRMAX(JJ),JJ=1,7)
    9 IMAX=50
      COORDM=DABS(COORD)
      RTDEG=180./3.14159265
C
C       LABEL= IDENTIFYING COMMENTS TO APPEAR WITH DATA PRINT-OUT
C       TZERO= INITIAL TIME
C        TINT=INTERVAL BETWEEN DESIRED DATA POINTS (SEC.)
C        TINCR=INTEGRATION STEP SIZE (SEC.)
C        DLAT= LATITUDE OF RADAR SITE (IN DEGREES)
C        NDATA=NUMBER OF DESIRED DATA POINTS
C       PRNT=PRINT-OUT SELECTOR
C          PRNT=-1. MINIMUM PRINT-OUT, TABULATED SUMMARY ONLY
C          PRNT= 0. PRINT-OUT OF COVARIANCE MATRICES + TABULATED SUMMARY
C          PRNT= 1. FULL PRINT-OUT
C       KLAMP=MEMORY OF ALGORITHM,EXPRESSED AS NUMBER OF DATA POINTS
C       MDATA=MODE SELECTOR
C          MDATA GREATER THAN 0.  TRAJECTORY PARAMETER ESTIMATION
C       COORD = COORDINATE AND UNIT SELECTOR
C          COORD=-2. POLAR COORDINATES, METRIC UNITS
C          COORD=+2. POLAR COORDINATES, ENGLISH UNITS
C       SIGMA = RMS NOISE COMPONENTS OF MEASUREMENT VECTOR
C       ERRMAX = TRAJECTORY INTEGRATION CONVERGENCE ERRORS (IN REC-
C          TANGULAR COORDINATES)
C
      WRITE (6,10)(LABEL(I),I=1,18)
   10 FORMAT('1',20X,'OUTPUT LISTING'/18A4//)
      WRITE(6,110)DLAT,NDATA,TINT,TINCR
  110 FORMAT(      /5X,'RADAR LATITUDE',3X,F10.5,2X,'DEGREES'/5X,I6,5X,'
     1DATA POINTS SPACED',3X, F10.6 ,2X,'SEC. APART, INTEGRATION STEP',2
     2X,F10.6,2X,'SEC.'//)
```

TABLE B-II (Continued)

```
      IF(CCORD)14,16,16
   14 RE=2.0925738D7/3.2808333
      WRITE(6,15)
   15 FORMAT(/5X,'METRIC UNITS(METERS,KILOGRAMS,RADIANS,SECONDS) USED TH
     1ROUGHOUT PROGRAM'//)
      GO TO 18
   16 RE=2.0925738D7
      WRITE(6,17)
   17 FORMAT(/5X,'ENGLISH UNITS(FEET,POUNDS,RADIANS,SECONDS) USED THROUG
     1HOUT PROGRAM'//)
   18 DO 19 J=1,7
   19 STATEM(J)=0.
   75 IF(KLAMP)81,81,751
  751 WRITE(6,752) KLAMP
  752 FORMAT(/5X,'MEMORY TIME =',I6,' DATA POINTS'//)
   81 WRITE(6,12)(ERRMAX(JJ),JJ=1,7)
   12 FORMAT(/5X,'MAXIMUM INTEGRATION CONVERGENCE ERRORS(IN RECTANGULAR
     1COORDINATES'/7D17.8///)
      L=2
      LAST=0
C
C     DATA LOOP
C
      DO 1000 NN=1,NDATA
      NN=NN
      NNP1=NN+1
C
C     TEST FOR 1ST TIME THROUGH
C
      IF(NN-1)21,20,21
C
C     READ INPUT DATA CARDS
C
   20 READ(5,30) TIME(1),RGN(1),AZN(1),ELN(1),RAPN(1)
   30 FORMAT(D15.2,4D15.6)
      IF(TIME(1)-TZERO)20,200,200
C
C     SET INITIAL STATE ESTIMATE = INITIAL NOISY STATE
C
  200 RHAT(1)=RGN(1)
      AHAT(1)=AZN(1)
      EHAT(1)=ELN(1)
  201 READ(5,30)  TIME(2),RGN(2),AZN(2),ELN(2),RAPN(2)
      DT=TIME(2)-TIME(1)
      IF(DT-TINT)201,202,202
  202 APHAT(1)=(AZN(2)-AZN(1))/DT
      EPHAT(1)=(ELN(2)-ELN(1))/DT
      CCVAR(1)=SIGMA(1)
      CCVAR(2)=SIGMA(2)
      CCVAR(3)=SIGMA(3)
      CCVAR(5)=1.414*SIGMA(2)/DT
      CCVAR(6)=1.414*SIGMA(3)/DT
      CCVAR(7)=1.
      IF(MDATA-3)203,204,205
  203 WRITE(6,210)
  210 FORMAT(//5X,'*** THIS MAIN PROGRAM REQUIRES MDATA = 3 OR 4, QUIT')
      RETURN
  204 RPHAT(1)=(RGN(2)-RGN(1))/DT
      CCVAR(4)=1.414*SIGMA(1)/DT
      GO TO 206
```

TABLE B-II (Continued)

```
  205 RPHAT(1)=RAPN(1)
      COVAR(4)=SIGMA(4)
  206 ESTATE(7)=1.D-4
      BETAH(1)=1./ESTATE(7)
C
C     PRINT HEADER PAGE OF OUTPUT LISTING
C
      WRITE(6,13) MDATA
   13 FORMAT(/5X,'ONLY THE 1ST',I3,1X,'QUANTITIES OF THE STATE VECTOR'/
     15X,'CONSISTING OF R,AZ,EL,RDOT,AZDOT,ELDOT,1./BETA, ARE MEASURED')
      WRITE(6,77)
   77 FORMAT(/5X,'RMS NOISE LEVELS ASSOCIATED WITH MEASUREMENT VECTOR'/)
      WRITE (6,7)(SIGMA(J),J=1,6)
    7 FORMAT(/5X,'SIGMA(R)=',F10.2,5X,'SIGMA(AZ)=',F10.6,5X,'SIGMA(EL)='
     1,F10.6,5X,'SIGMA(RDOT)=',F10.2/5X,'SIGMA(AZDOT)=',F10.6,5X,'SIGMA(
     2ELDOT)=',F10.6/)
      WRITE(6,78)
   78 FORMAT(/5X,'RMS NOISE LEVELS ASSOCIATED WITH INITIAL STATE VECTOR'
     1/)
      WRITE(6,7)(COVAR(J),J=1,6)
      WRITE(6,79) COVAR(7)
   79 FORMAT(5X,'SIGMA(1/BETA)=',F10.3//)
C
C     SET UP INITIAL STATE IN BOTH RADAR POLAR AND XYZ COORDINATES
C
      CALL RTOXY(RHAT(1),AHAT(1),EHAT(1),RPHAT(1),APHAT(1),EPHAT(1),ESTA
     1TE(1),ESTATE(2),ESTATE(3),ESTATE(4),ESTATE(5),ESTATE(6))
      HNXT=DSQRT(ESTATE(1)**2+ESTATE(2)**2+(ESTATE(3)+RE)**2)-RE
      XHAT(1)=ESTATE(1)
      YHAT(1)=ESTATE(2)
      ZHAT(1)=ESTATE(3)
      XPHAT(1)=ESTATE(4)
      YPHAT(1)=ESTATE(5)
      ZPHAT(1)=ESTATE(6)
      ALPHA(1)=ESTATE(7)
      HEIGHT(1)=HNXT
      NXT=1
      WRITE(6,49)NN
      WRITE(6,50)TIME(NXT),RHAT(NXT),AHAT(NXT),EHAT(NXT),RPHAT(NXT),APHA
     1T(NXT),EPHAT(NXT),ESTATE(1),ESTATE(2),ESTATE(3),ESTATE(4),ESTATE(5
     2),ESTATE(6),HNXT,BETAH(NXT)
      K=1
      NXT=2
      INDEX=2
      GO TO 31
C
C     UPDATE SUBSCRIPTS
C
   21 K=MOD(NN,IMAX)
      NXT=K+1
      INDEX=NXT
      IF(K)23,23,24
   23 K=IMAX
C
C     READ NEXT DATA POINT
C
   24 READ(5,30,END=8000)TIME(NXT),RGN(NXT),AZN(NXT),ELN(NXT),RAPN(NXT)
      DT=TIME(NXT)-TLAST
      IF(DT-TINT)24,31,31
   31 TLAST=TIME(NXT)
```

TABLE B-II (Continued)

```
      TAU=DT
C
C      CALL TRAJGX TO OBTAIN NEXT PREDICTED DATA POINT
C
      CALL TRAJGX(ESTATE(1),ESTATE(2),ESTATE(3),ESTATE(4),ESTATE(5),ESTA
     1TE(6),BETAH(K),TAU,TINCR,ERRMAX,STATEP(1),STATEP(2),STATEP(3),STAT
     2EP(4),STATEP(5),STATEP(6),PBETA,PROD)
C
C      CONVERT NEXT PREDICTED DATA POINT TO RADAR POLAR COORDINATES
C
 5700 CALL XYTOR(STATEP(1),STATEP(2),STATEP(3),STATEP(4),STATEP(5),STATE
     1P(6),PRA,PAZ,PEL,PRAP,PAZP,PELP)
      PH=DSQRT(STATEP(1)**2+STATEP(2)**2+(STATEP(3)+RE)**2)-RE
      STATEP(7)=1./PBETA
      STATEM(1)=RGN(NXT)
      STATEM(2)=AZN(NXT)
      STATEM(3)=ELN(NXT)
      STATEM(4)=RAPN(NXT)
      IF(PRNT)5702,569,569
C
C      PRINT OUT NOISY MEASUREMENT
C      PRINT OUT PREDICTED STATE
C
  569 WRITE(6,49)NNP1
   49 FORMAT('1',5X,I5,' TH DATA POINT'//)
      WRITE(6,58) TIME(NXT),RGN(NXT),AZN(NXT),ELN(NXT),RAPN(NXT)
   58 FORMAT(/5X,'NOISY DATA'/5X,'TIME =',F10.4,5X,'RGN=',F10.2,3X,'AZN=
     1',F10.6,3X,'ELN=',F10.6,3X,'RAPN=',F10.2//)
      WRITE(6,570)
  570 FORMAT(/5X,'PREDICTED STATE       BASED ON PAST DATA ONLY')
      WRITE(6,50)TIME(NXT),PRA,PAZ,PEL,PRAP,PAZP,PELP,STATEP(1),STATEP(2
     1),STATEP(3),STATEP(4),STATEP(5),STATEP(6),PH,PBETA
      IF(PRNT)5702,5702,5701
C
C      PRINT OUT TRANSITION MATRIX
C
 5701 WRITE(6,51)((PROD(JJ,KK),KK=1,7),JJ=1,7)
   51 FORMAT(/5X,'TRANSITION MATRIX'/(7D17.8))
C
C      CALL ESTMAT TO COMBINE MEASUREMENT WITH PREDICTION TO OBTAIN
C       ESTIMATE OF STATE VECTOR
C
 5702 CALL ESTMAT(STATEM,STATEP,PROD,ESTATE,STDEVX,STDEVR)
C
C      STORE EXPECTED RMS ERRORS
C
      SIGR (NXT)=STDEVR(1)
      SIGA (NXT)=STDEVR(2)
      SIGE (NXT)=STDEVR(3)
      SIGRP(NXT)=STDEVR(4)
      SIGAP(NXT)=STDEVR(5)
      SIGEP(NXT)=STDEVR(6)
      SIGX (NXT)=STDEVX(1)
      SIGY (NXT)=STDEVX(2)
      SIGZ (NXT)=STDEVX(3)
      SIGXP(NXT)=STDEVX(4)
      SIGYP(NXT)=STDEVX(5)
      SIGZP(NXT)=STDEVX(6)
      SIGALP(NXT)=STDEVX(7)
C
```

TABLE B-II (Continued)

```
C      CONVERT ESTIMATED STATE TO RADAR POLAR COORDINATES
C      STORE ESTIMATED STATE VECTOR IN OUTPUT ARRAYS
C
 5793 CALL XYTOR(ESTATE(1),ESTATE(2),ESTATE(3),ESTATE(4),ESTATE(5),ESTAT
     1E(6),RHAT(NXT),AHAT(NXT),EHAT(NXT),RPHAT(NXT),APHAT(NXT),EPHAT(NXT
     2))
      XHAT(NXT)=ESTATE(1)
      YHAT(NXT)=ESTATE(2)
      ZHAT(NXT)=ESTATE(3)
      XPHAT(NXT)=ESTATE(4)
      YPHAT(NXT)=ESTATE(5)
      ZPHAT(NXT)=ESTATE(6)
      ALPHA(NXT)=ESTATE(7)
      BETAH(NXT)=1./ESTATE(7)
      HNXT=DSQRT(ESTATE(1)**2+ESTATE(2)**2+(ESTATE(3)+RE)**2)-RE
      HEIGHT(NXT)=HNXT
      IF(PRNT)581,579,579
C
C      PRINT OUT ESTIMATED STATE
C
  579 WRITE(6,580)
  580 FORMAT(/5X,'ESTIMATED POINT'/)
      WRITE(6,50)TIME(NXT),RHAT(NXT),AHAT(NXT),EHAT(NXT),RPHAT(NXT),APHA
     1T(NXT),EPHAT(NXT),ESTATE(1),ESTATE(2),ESTATE(3),ESTATE(4),ESTATE(5
     2),ESTATE(6),HNXT,BETAH(NXT)
   50 FORMAT(5X,'TIME =',F10.4/5X,'RA =',F15.2,3X,'AZ =',F10.6,3X,'EL ='
     1,F10.6,3X,'RAP=',F10.2,3X,'AZP=',F10.6,3X,'ELP=',F10.6/5X,'X  =',F
     210.2,3X,'Y  =',F10.2,3X,'Z  =',F10.2,3X,'XP =',F10.2,3X,'YP =',F10
     3.2,3X,'ZP =',F10.2/5X,'HEIGHT =',5X,F15.2,20X,'BETA=',5X,F10.2//)
C
C       COMPUTE DIFFERENCES BETWEEN ESTIMATE AND MEASURED DATA POINT
C
  581 DR(NXT)=RHAT(NXT)-RGN(NXT)
      DA(NXT)=AHAT(NXT)-AZN(NXT)
      DE(NXT)=EHAT(NXT)-ELN(NXT)
      DRP(NXT)=RPHAT(NXT)-RAPN(NXT)
   25 IF(NN-NDATA)27,26,26
   26 KMAX=INDEX
      GO TO 40
   27 IF(INDEX-IMAX)1000,28,28
   28 KMAX=IMAX
C
C      WRITE OUTPUT
C
   40 WRITE(6,10)(LABEL(I),I=1,18)
  420 WRITE(6,43)
   43 FORMAT(5X,'MEASUREMENT DATA'//6X,'TIME',8X,'RANGE',7X,'RADOT',17X,
     1'AZIM',20X,'ELEV'//)
      WRITE(6,44)(TIME(KK),RGN(KK),RAPN(KK),AZN(KK),ELN(KK),KK=1,KMAX)
   44 FORMAT(3X,F10.3,2X,F10.2,2X,F10.2,14X,F10.6,14X,F10.6)
      WRITE(6,10)(LABEL(I),I=1,18)
      WRITE(6,60)
   60 FORMAT(     5X,'ESTIMATED VALUES'//)
      WRITE(6,41)
   41 FORMAT(     3X,'TIME',8X,'RANGE',7X,'RADOT',7X,'AZIM',8X,'AZDOT',7X
     1,'ELEV',8X, 'ELDOT',7X,'HEIGHT',6X,'BETA'//)
      WRITE(6,62)(TIME(J),RHAT(J),RPHAT(J),AHAT(J),APHAT(J),EHAT(J),EPHA
     1T(J),HEIGHT(J),BETAH(J),J=1,KMAX)
   62 FORMAT(3X,F10.3,2X,F10.2,1X,F10.2,2X,F10.6,2X,F10.6,2X,F10.6,2X,F1
     10.6,2X,F10.2,2X,F10.2)
```

TABLE B-II (Continued)

```
      WRITE(6,10)(LABEL(I),I=1,18)
      WRITE(6,60)
      WRITE(6,413)
      WRITE(6,414)(TIME(J),XHAT(J),YHAT(J),ZHAT(J),XPHAT(J),YPHAT(J),ZPH
     1AT(J),BETAH(J),ALPHA(J),J=1,KMAX)
  414 FORMAT(3X,F10.3,2X,F10.2,2X,F10.2,2X,F10.2,2X,F10.2,2X,F10.2,2X,F1
     10.2,2X,F10.2,2X,F12.8)
      WRITE(6,10)(LABEL(I),I=1,18)
      WRITE(6,63)
   63 FORMAT(5X,'RESIDUALS=(ESTIMATED VALUES)-(NOISY DATA)'//)
      WRITE(6,64)
   64 FORMAT(6X,'TIME',8X,'RANGE',7X,'RADOT',17X,'AZIM',20X,'ELEV'//)
      WRITE(6,44)(TIME(J),DR(J),DRP(J),DA(J),DE(J),J=L,KMAX)
  900 WRITE(6,10)(LABEL(I),I=1,18)
      WRITE(6,901)
  901 FORMAT(5X,'EXPECTED RMS ERRORS'//)
      WRITE(6,61)
   61 FORMAT(3X,'TIME',8X,'RANGE',7X,'RADOT',7X,'AZIM',8X,'AZDOT',7X,'EL
     1EV',8X,'ELDOT',7X,'8ETA',8X,'ALPHA'//)
      WRITE(6,902)(TIME(J),SIGR(J),SIGRP(J),SIGA(J),SIGAP(J),SIGE(J),SIG
     1EP(J),SIGALP(J),J=L,KMAX)
  902 FORMAT(3X,F10.3,2X,F10.3,2X,F10.2,2X,F10.6,2X,F10.6,2X,F10.6,2X,F1
     10.6,14X,F12.8)
      WRITE(6,10)(LABEL(I),I=1,18)
      WRITE(6,901)
      WRITE(6,413)
  413 FORMAT(3X,'TIME',8X,'X',11X,'Y',11X,'Z',11X,'XDOT',8X,'YDOT',8X,'Z
     1DOT',8X,'BETA',8X,'ALPHA'//)
      WRITE(6,903)(TIME(J),SIGX(J),SIGY(J),SIGZ(J),SIGXP(J),SIGYP(J),SIG
     1ZP(J),SIGALP(J),J=L,KMAX)
  903 FORMAT(3X,F10.3,2X,F10.2,2X,F10.2,2X,F10.2,2X,F10.2,2X,F10.2,2X,F1
     10.2,14X,F12.8)
      L=1
      IF(LAST)9000,1000,9000
 1000 CONTINUE
 1001 GO TO 1
 8000 IF(NXT-1)9000,9000,8001
 8001 KMAX=NXT-1
      LAST=1
      GO TO 40
 9000 RETURN
      END
```

TABLE B-III

SUBROUTINE TRAJGX

```
      SUBROUTINE TRAJGX(X1,Y1,Z1,XDOT1,YDOT1,ZDOT1,8ETA1,TAU ,TSTEP,ERRM
     1AX,X2,Y2,Z2,XDOT2,YDOT2,ZDOT2,8ETA2,PHIMAT)
      IMPLICIT REAL*8(A-H,O-Z)
      REAL*8 TAU,TSTEP
      REAL*8 J,K,L,M,N,X1,Y1,Z1,XDOT1,YDOT1,ZDOT1,8ETA1,TINT,TINCR,ERRMA
     1X,X2,Y2,Z2,XDOT2,YDOT2,ZDOT2,8ETA2,PHIMAT
      COMMON /FCOM/COORD,DLAT,PRNT/ICOM/KLAMP,MDATA,NN
      DIMENSION XVCTR(7,3),8VCTR(7,3),XI(7,3),ERR(7,3),AMTRX(7,7,3),ASQ(
     17,7,3),A8(7,3),PHI(7,7,3),UNIT(7,7),ERRMAG(7),ERRMAX(7),PHIMAT(7,7
     2)
C
C       CHECK THAT DATA INTERVAL,TAU, AND INTEGRATION STEP,TSTEP, HAVE
C       THE SAME SIGN
C
      TPROD=TAU*TSTEP
      IF(TPROD)1,2,2
    1 TINCR=-TSTEP
      WRITE(6,11)
   11 FORMAT(/5X,'DATA INTERVAL,TAU, AND INTEGRATION STEP,TSTEP, HAD OP-
     1POSITE SIGN.'/5X,'LATTER SIGN WAS CHANGED TO FORCE AGREEMENT'//)
      GO TO 5
    2 TINCR=TSTEP
    5 TM=DA8S(TAU)
      TSTM=DA8S(TSTEP)
      IF(TM-TSTM)3,4,4
    3 TINCR=TAU
    4 TINT=TAU
C
C     CHECK FOR IST DATA POINT
C
      IF(NN  -1)100,100,89
C
C     SET UP CONSTANTS AND ARRAYS
C
  100 PI=3.14159265
      RTDEG=1.8D2/PI
      OMEGA=7.27D-5
      RLAT=DLAT/RTDEG
      OMC=OMEGA*DCOS(RLAT)
      OMS=OMEGA*DSIN(RLAT)
      IF(COORD)1000,1001,1001
 1000 RE=2.0925738D7/3.2808333
      GM=1.40763488D16/(3.2808333)**3
      GAMMA=1.DO/7.D3
      GO TO 1002
 1001 RE=2.0925738D7
      GM=1.40763488D16
      GAMMA=.3048DO/7.D3
 1002 DO 101 JJ=1,7
      DO 101 KK=1,7
  101 UNIT(JJ,KK)=0.
      DO 102 JJ=1,7
      ERR(JJ,1)=0.
      ERR(JJ,2)=0.
      ERR(JJ,3)=0.
  102 UNIT(JJ,JJ)=1.
   89 IF(TAU)103,90,103
C
C     DATA INTERVAL,TAU, = 0.
C     SET OUTPUT STATE = INPUT STATE, TRANSITION MATRIX=IDENTITY MATRIX
C
   90 X2=X1
      Y2=Y1
```

TABLE B-III (Continued)

```
      Z2=Z1
      XOOT2=XDOT1
      YOOT2=YDOT1
      ZOOT2=ZOOT1
      BETA2=BETA1
      DO 91 JJ=1,7
      DO 91 KK=1,7
   91 PHIMAT(JJ,KK)=UNIT(JJ,KK)
      RETURN
C
C     SET INDICES
C
  103 NPOINT=1
      KOUNT=1
      DT=TINCR
      NEXT=2
      NLAST=3
      T=0.
      KOELT=0
      PHSKIP=0.
      TSQ=TINCR**2
C
C     SET UP INITIAL STATE VECTOR AND TRANSITION MATRIX
C
      XVCTR(1,1)=X1
      XVCTR(2,1)=Y1
      XVCTR(3,1)=Z1
      XVCTR(4,1)=XDOT1
      XVCTR(5,1)=YOOT1
      XVCTR(6,1)=ZOOT1
      XVCTR(7,1)=1./BETA1
      DO 104 JJ=1,7
      DO 104 KK=1,7
  104 PHI(JJ,KK,1)=UNIT(JJ,KK)
  300 CONTINUE
C
C     CALCULATE DRAG VELOCITY AND HEIGHT
C
      VO=DSQRT(XVCTR(4,NPOINT)**2+XVCTR(5,NPOINT)**2+XVCTR(6,NPOINT)**2)
      R=DSQRT(XVCTR(1,NPOINT)**2+XVCTR(2,NPOINT)**2+(XVCTR(3,NPOINT)+RE)
     1**2)
      HT=R-RE
      IF(HT)3001,3003,3003
C
C     NEGATIVE HEIGHT, APPARENT EARTH IMPACT, RETURN TO CALLING PROGRAM
C
 3001 WRITE(6,3002)HT,T,(XVCTR(JJ,NPOINT),JJ=1,7)
 3002 FORMAT(/5X,'HEIGHT=',F10.2,' ,AFTER ',F10.5,' SEC. OF INTEGRATION
     1-- APPARENT EARTH IMPACT'/5X,'STATE VECTOR='/7017.8/ 5X,'RETURNED
     2TO CALLING PROGRAM'///)
      NEXT=NPOINT
      GO TO 600
 3003 HKMFT=HT/1.03
      RHO=0.
      CALL ATM(HKMFT,RHO)
      ALPHA=XVCTR(7,NPOINT)
C
C     SET UP EQUATIONS OF MOTION
C
      A=-RHO*VO/2.DO
      W=GM/R**3
      B=OMEGA**2-W
      C=2.DO*OMS
```

TABLE B-III (Continued)

```
      E=-2.D0*OMC
      G=OMS**2-W
      J=-OMS*OMC
      K=J*RE
      N=OMC**2-W
      Q=N*RE
      XPP=A*XVCTR(7,NPOINT)*XVCTR(4,NPOINT)+B*XVCTR(1,NPOINT)+C*XVCTR(5,
     1NPOINT)+E*XVCTR(6,NPOINT)
      YPP=-C*XVCTR(4,NPOINT)+G*XVCTR(2,NPOINT)+A*XVCTR(7,NPOINT)*XVCTR(5
     1,NPOINT)+J*XVCTR(3,NPOINT)+K
      ZPP=-E*XVCTR(4,NPOINT)+J*XVCTR(2,NPOINT)+N*XVCTR(3,NPOINT)+A*XVCTR
     1(7,NPOINT)*XVCTR(6,NPOINT)+Q
C
C     FILL IN STATE VECTOR MATRICES
C
      BVCTR(1,NPOINT)=XVCTR(4,NPOINT)
      BVCTR(2,NPOINT)=XVCTR(5,NPOINT)
      BVCTR(3,NPOINT)=XVCTR(6,NPOINT)
      BVCTR(4,NPOINT)=XPP
      BVCTR(5,NPOINT)=YPP
      BVCTR(6,NPOINT)=ZPP
      BVCTR(7,NPOINT)=0.
C
C     CALCULATE SYSTEM MATRIX A
C
      DO 301 JJ=1,3
      DO 301 KK=1,7
  301 AMTRX(JJ,KK,NPOINT)=0.
      AMTRX(1,4,NPOINT)=1.
      AMTRX(2,5,NPOINT)=1.
      AMTRX(3,6,NPOINT)=1.
C
C     CALCULATE DERIVATIVES OF COEFFICIENTS
C
      DA=-A*GAMMA/R
      DADX=DA*XVCTR(1,NPOINT)
      DADY=DA*XVCTR(2,NPOINT)
      DADZ=DA*(XVCTR(3,NPOINT)+RE)
      IF(VD)701,702,701
  701 DAP=A/VD**2
      GO TO 703
  702 DAP=A/1.D-4
  703 CONTINUE
      DADXP=DAP*XVCTR(4,NPOINT)
      DADYP=DAP*XVCTR(5,NPOINT)
      DADZP=DAP*XVCTR(6,NPOINT)
      DB=-3.D0*W/R**3
      DBDX=DB*XVCTR(1,NPOINT)
      DBDY=DB*XVCTR(2,NPOINT)
      DBDZ=DB*(XVCTR(3,NPOINT)+RE)
      DGDX=DBDX
      DGDY=DBDY
      DGDZ=DBDZ
      DNDX=DBDX
      DNDY=DBDY
      DNDZ=DBDZ
      DQDX=RE*DBDX
      DQDY=RE*DBDY
      DQDZ=RE*DBDZ
C
C     FOR CONSTANT BETA
C
      DALFDX=0.
```

TABLE B-III (Continued)

```
      DALFDY=0.
      DALFDZ=0.
C
      DALADX=ALPHA*DADX+A*DALFDX
      DALADY=ALPHA*DADY+A*DALFDY
      DALADZ=ALPHA*DADZ+A*DALFDZ
      AMTRX(4,1,NPOINT)=B+XVCTR(1,NPOINT)*DBDX+XVCTR(4,NPOINT)*DALADX
      AMTRX(4,2,NPOINT)=XVCTR(1,NPOINT)*DBDY+XVCTR(4,NPOINT)*DALADY
      AMTRX(4,3,NPOINT)=XVCTR(1,NPOINT)*DBDZ+XVCTR(4,NPOINT)*DALADZ
      AMTRX(4,4,NPOINT)=ALPHA*(A+XVCTR(4,NPOINT)*DADXP)
      AMTRX(4,5,NPOINT)=XVCTR(4,NPOINT)*ALPHA*DADYP+C
      AMTRX(4,6,NPOINT)=XVCTR(4,NPOINT)*ALPHA*DADZP+E
      AMTRX(4,7,NPOINT)=A*XVCTR(4,NPOINT)
      AMTRX(5,1,NPOINT)=XVCTR(2,NPOINT)*DGDX+XVCTR(5,NPOINT)*DALADX
      AMTRX(5,2,NPOINT)=G+XVCTR(2,NPOINT)*DGDY+XVCTR(5,NPOINT)*DALADY
      AMTRX(5,3,NPOINT)=J+XVCTR(2,NPOINT)*DGDZ+XVCTR(5,NPOINT)*DALADZ
      AMTRX(5,4,NPOINT)=XVCTR(5,NPOINT)*ALPHA*DADXP-C
      AMTRX(5,5,NPOINT)=ALPHA*(A+XVCTR(5,NPOINT)*DADYP)
      AMTRX(5,6,NPOINT)=XVCTR(5,NPOINT)*ALPHA*DADZP
      AMTRX(5,7,NPOINT)=A*XVCTR(5,NPOINT)
      AMTRX(6,1,NPOINT)=DQDX+XVCTR(3,NPOINT)*DNDX+XVCTR(6,NPOINT)*DALADX
      AMTRX(6,2,NPOINT)=J+DQDY+XVCTR(3,NPOINT)*DNDY+XVCTR(6,NPOINT)*DALA
     1DY
      AMTRX(6,3,NPOINT)=N+XVCTR(3,NPOINT)*DNDZ+DQDZ+XVCTR(6,NPOINT)*DALA
     1DZ
      AMTRX(6,4,NPOINT)=XVCTR(6,NPOINT)*ALPHA*DADXP-E
      AMTRX(6,5,NPOINT)=XVCTR(6,NPOINT)*ALPHA*DADYP
      AMTRX(6,6,NPOINT)=ALPHA*(A+XVCTR(6,NPOINT)*DADZP)
      AMTRX(6,7,NPOINT)=A*XVCTR(6,NPOINT)
      AMTRX(7,1,NPOINT)=DALFDX
      AMTRX(7,2,NPOINT)=DALFDY
      AMTRX(7,3,NPOINT)=DALFDZ
      DO 3010 JJ=4,7
 3010 AMTRX(7,JJ,NPOINT)=0.
      DO 302 JJ=1,7
      AB(JJ,NPOINT)=0.
      DO 302 LL=1,7
  302 AB(JJ,NPOINT)=AB(JJ,NPOINT)+AMTRX(JJ,LL,NPOINT)*BVCTR(LL,NPOINT)
      IF(PHSKIP)400,303,400
C
C     CALCULATE TRANSITION MATRIX
C
  303 DO 304 JJ=1,7
      DO 304 KK=1,7
      ASQ(JJ,KK,NPOINT)=0.
      DO 304 LL=1,7
  304 ASQ(JJ,KK,NPOINT)=ASQ(JJ,KK,NPOINT)+AMTRX(JJ,LL,NPOINT)*AMTRX(LL,
     1KK,NPOINT)
      KPOINT=NLAST
      DO 305 JJ=1,7
      DO 305 KK=1,7
  305 PHI(JJ,KK,KPOINT)=UNIT(JJ,KK)+TINCR*AMTRX(JJ,KK,NPOINT)+TSQ*ASQ(JJ
     1,KK,NPOINT)/2.00
      DO 306 JJ=1,7
      DO 306 KK=1,7
      PHI(JJ,KK,NEXT)=0.
      DO 306 LL=1,7
  306 PHI(JJ,KK,NEXT)=PHI(JJ,KK,NEXT)+PHI(JJ,LL,KPOINT)*PHI(LL,KK,NPOINT
     1)
C
C     INTEGRATE OVER INTERVAL TINCR
C
      IF(KOELT*KOUNT)320,320,330
```

TABLE B-III (Continued)

```
  320 DO 321 JJ=1,7
  321 XI(JJ,NEXT)=XVCTR(JJ,NPOINT)+TINCR*BVCTR(JJ,NPOINT)+TSQ*AB(JJ,NPOI
     1NT)/2.00
      GO TO 350
  330 DO 331 JJ=1,7
  331 XI(JJ,NEXT)=XVCTR(JJ,NLAST)+2.00*TINCR*BVCTR(JJ,NLAST)+2.00/3.D0*T
     1SQ*(AB(JJ,NLAST)+2.D0*AB(JJ,NPOINT))
  350 NPT=NPOINT
      NPOINT=NEXT
      DO 351 JJ=1,7
  351 XVCTR(JJ,NPOINT)=XI(JJ,NEXT)
      PHSKIP=1.
      GO TO 300
  400 DO 4000 JJ=1,7
      XVCTR(JJ,NEXT)=XVCTR(JJ,NPT)+.5D0*TINCR*(BVCTR(JJ,NPT)+BVCTR(JJ,NP
     1OINT))+TSQ*(AB(JJ,NPT)-AB(JJ,NEXT))/1.201
      ERR(JJ,NEXT)=XI(JJ,NEXT)-XVCTR(JJ,NEXT)
      ERRMAG(JJ)=DABS(ERR(JJ,NEXT))
 4000 CONTINUE
C
C     TEST INTEGRATION CONVERGENCE ERRORS
C
      DO 401 JJ=1,7
      IF(ERRMAG(JJ)-ERRMAX(JJ))401,500,500
  401 CONTINUE
      T=T+TINCR
      TREM=TAU-T
      IF(TREM)4010,600,4010
 4010 TREMAG=DABS(TREM)
      IF(TREMAG-TSTM)405,402,402
  405 TINCR=TREM
      TSQ=TINCR**2
      KOUNT=0
C
C     UPDATE SUBSCRIPTS
C
  402 NEXT=MOD(NEXT,3)+1
      NLAST=NPT
      KDELT=KDELT+1
      PHSKIP=0.
      GO TO 300
C
C     CONVERGENCE ERRORS TOO LARGE -- PERFORM ITERATION
C
  500 DO 501 KK=1,7
  501 XI(KK,NEXT)=XVCTR(KK,NEXT)
      PHSKIP=1.
      GO TO 300
C
C     SET UP OUTPUT STATE VECTOR AND TRANSITION MATRIX
C
  600 X2=XVCTR(1,NEXT)
      Y2=XVCTR(2,NEXT)
      Z2=XVCTR(3,NEXT)
      XDOT2=XVCTR(4,NEXT)
      YDOT2=XVCTR(5,NEXT)
      ZDOT2=XVCTR(6,NEXT)
      BETA2=1./XVCTR(7,NEXT)
      DO 601 JJ=1,7
      DO 601 KK=1,7
  601 PHIMAT(JJ,KK)=PHI(JJ,KK,NEXT)
      RETURN
      END
```

TABLE B-IV

SUBROUTINE ESTMAT (VERSION I)

```
      SUBROUTINE ESTMAT(STATEM,STATEP,PHIMTX,ESTATE,DEVX,DEVR)
C
C     SUBROUTINE ESTMAT USING EQUATIONS IN MOWERY ARTICLE
C
      IMPLICIT REAL*8(A-H,O-Z)
      REAL*8 STATEM,STATEP,PHIMTX,ESTATE
      COMMON /ACOM/CCVAR(7),SIGMA(7)/FCOM/COORD,DLAT,PRNT/ICOM/KLAMP,MDA
     1TA,NN
      DIMENSION PHIMTX(7,7),DUMMY(7,7),          W(7,7),DX(7),STATEP(7)
      DIMENSION STATEM(7),ESTATE(7),H(7,7),R(7,7),S(7,7,2)
      DIMENSION DEVX(7),DEVR(7)
      DIMENSION DU77(7,7),SR(7,7,2),PHIT(7,7),UNIT(7,7)
      DIMENSION HT(7,7),RSTATE(7),MV1(7),MV2(7),ARRAY(49)
      NZERO=0
      COORDM=DABS(COORD)
C
C     TEST FOR 1ST TIME THROUGH
C
      IF(NN-1)1,1,100
C
C     1ST TIME THROUGH - INITIALIZE ARRAYS AND MATRICES
C     SET UP APRIORI COVARIANCE MATRIX
C
    1 NSTART=1
      IF(MDATA)2,200,3
    2 MD=-MDATA
      GO TO 4
    3 MD=MDATA
    4 MP1=MD+1
      LMAX=MD**2
      NK=1
      NKM1=2
      DO 7 J=1,7
      DO 7 K=1,7
      UNIT(J,K)=0.
      W(J,K)=0.
      DX(J)=0.
      P(J,K)=0.
    5 R(J,K)=0.
      SR(J,K,1)=0.
    7 S(J,K,1)=0.
      DO 6 J=1,MD
      P(J,J)=1.
    6 R(J,J)=SIGMA(J)**2
   88 DO 8 J=1,7
    8 UNIT(J,J)=1.
      IF(COORDM-1.)90,90,80
   90 DO 91 J=1,7
   91 S(J,J,1)=CCVAR(J)**2
      GO TO 1302
   80 DO 81 J=1,7
   81 SR(J,J,1)=CCVAR(J)**2
      IF(PRNT)10,9,9
    9 WRITE(6,1180)((SR(J,K,NK),K=1,7),J=1,7)
 1180 FORMAT(/5X,'APRIORI COVARIANCE MATRIX IN RADAR COORDINATES'/(7D17.
     18))
   10 DO 1181 J=1,7
      DO 1181 K=1,7
 1181 DU77(J,K)=SR(J,K,NK)
      CALL RCOVTX(DU77,ESTATE,DUMMY)
```

TABLE B-IV (Continued)

```
      DO 1182 J=1,7
      DO 1182 K=1,7
 1182 S(J,K,NK)=DUMMY(J,K)
      GO TO 1302
C
C     UPDATE INDICES NK,NKM1
C
  100 IF(NK-1)1,102,103
  102 NK=2
      NKM1=1
      GO TO 104
  103 NK=1
      NKM1=2
C
C     COMPUTE COVARIANCE MATRIX S(K,K-1) IN XYZ COORDINATES
C
  104 DO 105 J=1,7
      DO 105 K=1,7
      PHIT(J,K)=PHIMTX(K,J)
  105 DUMMY(J,K)=S(J,K,NKM1)
      CALL DMTMUL(DUMMY,PHIT,DU77,7,7,7)
      CALL DMTMUL(PHIMTX,DU77,DUMMY,7,7,7)
      NSTART=0
      DO 106 J=1,7
      DO 106 K=1,7
  106 S(J,K,NKM1)=DUMMY(J,K)
      IF(PRNT)1060,1060,1058
C
C     PRINT OUT TRANSITION MATRIX
C     PRINT OUT COVARIANCE MATRIX S(K,K-1) OF PREDICTED STATE
C
 1058 WRITE(6,300)((PHIMTX(J,K),K=1,7),J=1,7)
  300 FORMAT(/5X,'TRANSITION MATRIX USED IN ESTIMATION'/(7D17.8))
      WRITE(6,1059)
 1059 FORMAT(/5X,'COVARIANCE MATRICES S(K,K-1)')
      WRITE (6,1183)((S(J,K,NKM1),K=1,7),J=1,7)
C
C     CALCULATE HYBRID COVARIANCE MATRIX OF PREDICTED STATE
C
      CALL XCOVTR(DUMMY,STATEP,DU77,7)
      DO 2016 J=1,7
      DO 2015 K=1,7
 2015 H(J,K)=DU77(J,K)/DSQRT(DU77(J,J)*DU77(K,K))
 2016 H(J,J)=DSQRT(DU77(J,J))
C
C     PRINT OUT HYBRID COVARIANCE MATRIX OF PREDICTED STATE
C
      WRITE(6,1301)((H(J,K),K=1,7),J=1,7)
C
C     COMPUTE WEIGHTING MATRIX W
C
 1060 CALL XCOVTR(DUMMY,STATEP,H,-1)
      DO 107 J=1,7
      DO 107 K=1,7
  107 HT(J,K)=H(K,J)
      CALL DMTMUL(DUMMY,HT,DU77,7,7,MD)
      CALL DMTMUL(H,DU77,DUMMY,MD,7,MD)
      DO 108 J=1,MD
      DO 108 K=1,MD
  108 DU77(J,K)=DUMMY(J,K)+R(J,K)
```

TABLE B-IV (Continued)

```
      DO 109 J=MP1,7
      DO 109 K=MP1,7
  109 DU77(J,K)=0.
      DO 1090 J=1,MD
      DO 1090 K=1,MD
      L=MD*(J-1)+K
 1090 ARRAY(L)=DU77(J,K)
      CALL MINV(ARRAY,MD,DETERM,MV1,MV2)
      DO 1091 L=1,LMAX
      J=(L-1)/MD+1
      K=L-MD*(J-1)
 1091 DU77(J,K)=ARRAY(L)
      CALL DMTMUL(HT,DU77,DUMMY,7,MD,MD)
      DO 110 J=1,7
      DO 110 K=1,7
  110 DU77(J,K)=S(J,K,NKM1)
      CALL DMTMUL(DU77,DUMMY,W,7,7,MD)
      IF(PRNT)1102,1102,2219
C
C     PRINT OUT PARTIAL DERIVATIVE MATRIX H=DR/DX, AND WEIGHTING MATRIX
C     EVALUATED AT PREDICTED STATE
C
 2219 WRITE(6,2220)((H(J,K),K=1,7),J=1,7)
 2220 FORMAT(/5X,'H MATRIX (DR/DX) EVALUATED AT PREDICTED STATE'/(7D17.8
     1))
 1101 WRITE(6,1111)((W(JJ,KK),KK=1,7),JJ=1,7)
 1111 FORMAT(/5X,'WEIGHTING MATRIX W'/(7D17.8))
C
C     CALCULATE COVARIANCE MATRIX S(K) OF ESTIMATED STATE
C
 1102 CALL DMTMUL(W,H,DUMMY,7,MD,7)
      DO 117 J=1,7
      DO 117 K=1,7
  117 DUMMY(J,K)=UNIT(J,K)-DUMMY(J,K)
      DO 118 J=1,7
      DO 118 K=1,7
      S (J,K,NK)=0.
      DO 118 L=1,7
  118 S (J,K,NK)=S (J,K,NK)+DUMMY(J,L)*S (L,K,NKM1)
      IF(MDATA)1056,200,1112
 1056 DO 1057 J=1,7
 1057 ESTATE(J)=STATEP(J)
      GO TO 1000
C
C     COMPUTE NEW ESTIMATE OF STATE VECTOR
C
 1112 CALL XYTOR(STATEP(1),STATEP(2),STATEP(3),STATEP(4),STATEP(5),STATE
     1P(6),RSTATE(1),RSTATE(2),RSTATE(3),RSTATE(4),RSTATE(5),RSTATE(6))
      RSTATE(7)=STATEP(7)
      DO 113 K=1,MD
  113 DX(K)=STATEM(K)-RSTATE(K)
      IF(PRNT)1132,1132,1130
 1130 WRITE(6,1131)(DX(J),J=1,7)
 1131 FORMAT(/5X,'Y-H(X)=',7D17.8)
 1132 DO 114 J=1,7
      DUMMY(J,1)=0.
      DO 114 K=1,MD
  114 DUMMY(J,1)=DUMMY(J,1)+W(J,K)*DX(K)
      DO 115 K=1,7
  115 ESTATE(K)=STATEP(K)+DUMMY(K,1)
```

TABLE B-IV (Continued)

```
C
C      CALCULATE HYBRID COVARIANCE MATRIX OF ESTIMATED STATE
C
 1000 DO 1190 J=1,7
      DO 1190 K=1,7
 1190 DUMMY(J,K)=S(J,K,NK)
      CALL XCOVTR(DUMMY,ESTATE,DU77,7)
      DO 1300 J=1,7
      DO 1297 K=1,7
      IF(DU77(J,J))1291,1291,1290
 1290 IF(DU77(K,K))1291,1291,1294
 1291 WRITE(6,1292)
 1292 FORMAT(/5X,'***NEGATIVE DIAGONAL TERMS OF COVARIANCE MATRIX IN RAD
     1AR POLAR COORDINATES***'/8X,'DISREGARD HYBRID MATRIX PRINTED BELOW
     2'//)
      PRNT=1.
      GO TO 1399
 1294 CONTINUE
      DUMMY(J,K)=DU77(J,K)/DSQRT(DU77(J,J)*DU77(K,K))
      CORR=DABS(DUMMY(J,K))
      IF(CORR-1.00001)1297,1297,1298
 1298 WRITE(6,1299)
 1299 FORMAT(///5X,'****** CORRELATION COEFFICIENT EXCEEDS 1. ******'//)
      PRNT=1.
 1297 CONTINUE
 1300 DUMMY(J,J)=DSQRT(DU77(J,J))
C
C      STORE DIAGONAL ELEMENTS OF COVARIANCE MATRICES
C
 1399 DO 1400 J=1,7
      DEVR(J)=DUMMY(J,J)
 1400 DEVX(J)=DSQRT(S(J,J,NK))
      IF(PRNT)1302,1401,1401
C
C      PRINT OUT STATE ESTIMATE AND COVARIANCE MATRICES
C
 1401 WRITE(6,1150) (ESTATE(J),J=1,7)
 1150 FORMAT(//5X,'ESTATE = ESIMATE OF STATE VECTOR'/7D17.8)
 1001 WRITE(6,1179)
 1179 FORMAT(/5X,'FINAL ESTIMATE OF COVARIANCE MATRICES')
      WRITE(6,1183)((S(J,K,NK),K=1,7),J=1,7)
 1183 FORMAT(//5X,'COVARIANCE MATRIX IN XYZ COORDINATES'/(7D17.8))
13001 WRITE(6,1301)((DUMMY(J,K),K=1,7),J=1,7)
 1301 FORMAT(/5X,'HYBRID MATRIX IN RADAR COORDINATES'/5X, DIAGONAL TERMS
     1 ARE STANDARD DEVIATIONS, OFF-DIAGONAL TERMS ARE CORRELATION COEFF
     2ICIENTS'/(7D17.8))
C
C      TEST FOR MEMORY CLAMPING
C
 1302 IF(KLAMP)1055,1055,119
  119 IF(NN-KLAMP)1055,1191,1191
 1191 DO 1192 J=1,7
      DO 1192 K=1,7
      L=7*(J-1)+K
 1192 ARRAY(L)=S(J,K,NK)
      CALL MINV(ARRAY,7,DET,MV1,MV2)
      IF(NN-KLAMP)1055,1193,1194
 1193 DELAST=DET
      GO TO 1055
C
```

TABLE B-IV (Continued)

```
C      SCALE COVARIANCE MATRIX
C
 1194 SCALE=(DELAST/DET)**.142857143
      DO 1195 J=1,7
      DO 1195 K=1,7
 1195 S(J,K,NK)=SCALE*S(J,K,NK)
 1055 IF(NSTART)100,1200,100
 1200 RETURN
  200 WRITE (6,201)
  201 FORMAT(///'****** MDATA=0, THIS SUBROUTINE SHOULD NOT HAVE BEEN CA
     1LLED ****'////)
      RETURN
      END
```

TABLE B-V

SUBROUTINE ESTMAT (VERSION II)

```
      SUBROUTINE ESTMAT(STATEM,STATEP,PHIMTX,ESTATE,DEVX,DEVR)
C
C     THIS VERSION OF ESTMAT INVERTS 7X7 COVARIANCE MATRICES
C     IN THE COURSE OF CALCULATIONS
C
      IMPLICIT REAL*8(A-H,O-Z)
      REAL*8 STATEM,STATEP,PHIMTX,ESTATE
      COMMON /ACOM/COVAR(7),SIGMA(7)/FCOM/COORD,DLAT,PRNT/ICOM/KLAMP,MDA
     1TA,NN
      DIMENSION PHIMTX(7,7),DUMMY(7,7),DUM(7,7) ,W(7,7),DX(7),STATEP(7)
      DIMENSION STATEM(7),ESTATE(7),H(7,7),R(7,7),S(7,7,2),RSTATE(7)
      DIMENSION  RINV(7,7),SI(7,7,2),HT(7,7),HBACK(7,7),HTBACK(7,7)
      DIMENSION DU77(7,7),MV1(7),MV2(7),ARRAY(49) ,SR(7,7,2),PHIT(7,7)
      DIMENSION DEVX(7),DEVR(7)
      NZERO=0
      COORDM=DABS(COORD)
C
C     TEST FOR 1ST TIME THROUGH
C
      IF(NN-1)1,1,100
C
C     1ST TIME THROUGH - INITIALIZE ARRAYS AND MATRICES
C     SET UP APRIORI COVARIANCE MATRIX
C
    1 NSTART=1
      IF(MDATA)2,500,3
    2 MD=-MDATA
      GO TO 4
    3 MD=MDATA
    4 MPI=MD+1
      NK=1
      NKM1=2
      DO 7 J=1,7
      DX(J)=0.
      DO 7 K=1,7
      DUMMY(J,K)=0.
      DU77(J,K)=0.
      W(J,K)=0.
      H(J,K)=0.
      HBACK(J,K)=0.
      HTBACK(J,K)=0.
    5 R(J,K)=0.
      SR(J,K,1)=0.
      RINV(J,K)=0.
      SI(J,K,1)=0.
      SI(J,K,2)=0.
    7 S(J,K,1)=0.
      DO 6 J=1,MD
      RINV(J,J)=1./SIGMA(J)**2
    6 R(J,J)=SIGMA(J)**2
      IF(COORDM-1.)11,11,61
   61 DO 8 J=1,7
      DUMMY(J,J)=1./COVAR(J)**2
    8 SR(J,J,1)=COVAR(J)**2
      CALL XCOVTR(DUMMY,ESTATE,H,-1)
      DO 9 J=1,7
      DO 9 K=1,7
    9 HT(J,K)=H(K,J)
      CALL DMTMUL(HT,DUMMY,DU77,7,7,7)
      CALL DMTMUL(DU77,H,DUMMY,7,7,7)
```

TABLE B-V (Continued)

```
      DO 10 J=1,7
      DO 10 K=1,7
   10 SI(J,K,1)=DUMMY(J,K)
      IF(PRNT)1000,1001,1001
 1001 WRITE(6,1180)((SR(J,K,NK),K=1,7),J=1,7)
 1180 FORMAT(//5X,'COVARIANCE MATRIX IN RADAR COORDINATES'/(7D17.8))
      WRITE(6,2011)((DUMMY(J,K),K=1,7),J=1,7)
      GO TO 1000
   11 DO 12 J=1,7
      S(J,J,1)=COVAR(J)**2
      SI(J,J,1)=1./S(J,J,1)
   12 DU77(J,J)=S(J,J,1)
      IF(PRNT)1000,13,13
   13 CALL XCOVTR(DU77,ESTATE,DUMMY,7)
      DO 14 J=1,7
      DO 14 K=1,7
   14 SR(J,K,1)=DUMMY(J,K)
      WRITE(6,1180)((SR(J,K,NK),K=1,7),J=1,7)
      WRITE(6,2011)((SI(J,K,NK),K=1,7),J=1,7)
      GO TO 1000
C
C     UPDATE INDICES NK,NKM1
C
  100 IF(NK-1)1,102,103
  102 NK=2
      NKM1=1
      GO TO 104
  103 NK=1
      NKM1=2
  104 NSTART=0
      IF(PRNT)106,105,105
C
C     PRINT OUT TRANSITION MATRIX
C
  105 WRITE(6,200)((PHIMTX(J,K),K=1,7),J=1,7)
  200 FORMAT(/5X,'TRANSITION MATRIX USED IN ESTIMATION'/(7D17.8))
C
C     COMPUTE INVERSE COVARIANCE MATRIX SI(K,K-1) IN XYZ COORDINATES
C
  106 DO 2001 J=1,7
      DO 2001 K=1,7
      L=7*(J-1)+K
 2001 ARRAY(L)=PHIMTX(J,K)
      CALL MINV(ARRAY,7,DETERM,MV1,MV2)
      DO 2002 L=1,49
      J=(L-1)/7+1
      K=L-7*(J-1)
 2002 PHIMTX(J,K)=ARRAY(L)
      DO 201 J=1,7
      DO 201 K=1,7
      PHIT(J,K)=PHIMTX(K,J)
  201 DUMMY(J,K)=SI(J,K,NKM1)
      CALL DMTMUL(DUMMY,PHIMTX,DU77,7,7,7)
      CALL DMTMUL(PHIT,DU77,DUMMY,7,7,7)
 2030 CALL XCOVTR(RINV,STATEP,H,-1)
      DO 203 J=1,7
      DO 203 K=1,7
  203 HT(J,K)=H(K,J)
      CALL DMTMUL(HT,RINV,W,7,7,7)
      IF(PRNT)2039,2039,2010
```

TABLE B-V (Continued)

```
C
C      CALCULATE HYBRID COVARIANCE MATRIX OF PREDICTED STATE
C
 2010 DO 202 J=1,7
      DO 202 K=1,7
  202 DU77(J,K)=DUMMY(J,K)
      DO 2003 J=1,7
      DO 2003 K=1,7
      L=7*(J-1)+K
 2003 ARRAY(L)=DU77(J,K)
      CALL MINV(ARRAY,7,DETERM,MV1,MV2)
      DO 2004 L=1,49
      J=(L-1)/7+1
      K=L-7*(J-1)
 2004 DU77  (J,K)=ARRAY(L)
      CALL XCOVTR(DU77,STATEP,HBACK,7)
      DO 2016 J=1,7
      DO 2015 K=1,7
 2015 HTBACK(J,K)=HBACK(J,K)/DSQRT(HBACK(J,J)*HBACK(K,K))
 2016 HTBACK(J,J)=DSQRT(HBACK(J,J))
C
C      PRINT OUT COVARIANCE MATRIX S(K,K-1) OF PREDICTED STATE
C      AND ITS INVERSE
C
C      PRINT OUT HYBRID COVARIANCE MATRIX OF PREDICTED STATE
C
C      PRINT OUT PARTIAL DERIVATIVE MATRIX H=DR/DX, AND HTRANS*QINV MATRX
C      EVALUATED AT PREDICTED STATE
C
      WRITE (6,1059)
 1059 FORMAT(/5X,'COVARIANCE MATRICES S(K,K-1)')
      WRITE(6,2011)((DUMMY(J,K),K=1,7),J=1,7)
 2011 FORMAT(/5X,'INVERSE MATRIX IN XYZ COORDINATES'/(7D17.8))
      WRITE(6,1183)((DU77(J,K),K=1,7),J=1,7)
      WRITE(6,1301)((HTBACK(J,K),K=1,7),J=1,7)
      WRITE(6,2220)((H(J,K),K=1,7),J=1,7)
      WRITE(6,2221)((W(J,K),K=1,7),J=1,7)
 2220 FORMAT(/5X,'H MATRIX (DR/DX) EVALUATED AT PREDICTED STATE'/(7D17.8
     1))
 2221 FORMAT(/5X,'W MATRIX = HTRANSP*QRINV'/(7D17.8))
C
C      CALCULATE COVARIANCE MATRIX S(K) OF ESTIMATED STATE
C
 2039 CALL DMTMUL(W,H,DU77,7,7,7)
      DO 204 J=1,7
      DO 204 K=1,7
      DUMMY(J,K)=DUMMY(J,K)+DU77(J,K)
  204 SI(J,K,NK)=DUMMY(J,K)
 2050 DO 2005 J=1,7
      DO 2005 K=1,7
      L=7*(J-1)+K
 2005 ARRAY(L)=DUMMY(J,K)
      CALL MINV(ARRAY,7,DET,MV1,MV2)
      DO 2006 L=1,49
      J=(L-1)/7+1
      K=L-7*(J-1)
      S(J,K,NK)=ARRAY(L)
 2006 DUMMY (J,K)=ARRAY(L)
      CALL DMTMUL(DUMMY,W,DUM,7,7,7)
      IF(MDATA)1056,500 ,1058
```

TABLE B-V (Continued)

```
 1056 DO 1057 J=1,7
 1057 ESTATE(J)=STATEP(J)
      GO TO 1148
C
C     COMPUTE ESTIMATE OF STATE VECTOR
C
 1058 CALL XYTCR(STATEP(1),STATEP(2),STATEP(3),STATEP(4),STATEP(5),STATE
     1P(6),RSTATE(1),RSTATE(2),RSTATE(3),RSTATE(4),RSTATE(5),RSTATE(6))
      RSTATE(7)=STATEP(7)
      DO 113 K=1,MD
  113 DX(K)=STATEM(K)-RSTATE(K)
      IF(PRNT)1132,1132,1130
 1130 WRITE(6,1131)(DX(J),J=1,7)
 1131 FORMAT(5X,'Y-H(X)=',7D17.8)
 1132 DO 2061 J=1,7
      DU77(J,1)=0.
      DO 2061 L=1,MD
 2061 DU77(J,1)=DU77(J,1)+DUM(J,L)*DX(L)
      DO 208 J=1,7
  208 ESTATE(J)=STATEP(J)+DU77(J,1)
C
C     CALCULATE HYBRID COVARIANCE MATRIX OF ESTIMATED STATE
C
 1148 DO 1800 J=1,7
      DO 1800 K=1,7
 1800 DUMMY(J,K)=S(J,K,NK)
      CALL XCOVTR(DUMMY,ESTATE,DU77,7)
      DO 1300 J=1,7
      DO 1297 K=1,7
      IF(DU77(J,J))1291,1291,1290
 1290 IF(DU77(K,K))1291,1291,1294
 1291 WRITE(6,1292)
 1292 FORMAT(15X,'***NEGATIVE DIAGONAL TERMS OF COVARIANCE MATRIX IN RAD
     1AR POLAR COORDINATES***'/8X,'DISREGARD HYBRID MATRIX PRINTED BELOW
     2'//)
      PRNT=1.
      GO TO 1399
 1294 CONTINUE
      DUMMY(J,K)=DU77(J,K)/DSQRT(DU77(J,J)*DU77(K,K))
      CORR=DABS(DUMMY(J,K))
      IF(CORR-1.00001)1297,1297,1298
 1298 WRITE(6,1299)
 1299 FORMAT(///5X,'*****CORRELATION COEFFICIENT EXCEEDS 1. *****'///)
      PRNT=1.
 1297 CONTINUE
 1300 DUMMY(J,J)=DSQRT(DU77(J,J))
C
C     STORE DIAGONAL ELEMENTS OF COVARIANCE MATRICES
C
 1399 DO 1400 J=1,7
      DEVR(J)=DUMMY(J,J)
 1400 DEVX(J)=DSQRT(S(J,J,NK))
      IF(PRNT)1302,1149,1149
C
C     PRINT OUT STATE ESTIMATE AND COVARIANCE MATRICES
C
 1149 WRITE(6,1152)((DUM(J,K),K=1,7),J=1,7)
1152  FORMAT(/5X,'WEIGHTING MATRIX S(K)*W '/(7D17.8))
 1150 WRITE(6,1151) (ESTATE(J),J=1,7)
 1151 FORMAT(//5X,'ESTATE = ESIMATE OF STATE VECTOR'/7D17.8)
```

TABLE B-V (Continued)

```
      WRITE(6,1179)
 1179 FORMAT(//5X,'FINAL ESTIMATE OF COVARIANCE MATRICES')
      WRITE(6,1183)((S(J,K,NK),K=1,7),J=1,7)
      WRITE(6,2011)((SI(J,K,NK),K=1,7),J=1,7)
 1183 FORMAT(//5X,'COVARIANCE MATRIX IN XYZ COORDINATES'/(7D17.8))
      WRITE(6,1184) DET
 1184 FORMAT(/5X,'DETERMINANT=',D17.8//)
      WRITE(6,1301)((DUMMY(J,K),K=1,7),J=1,7)
 1301 FORMAT(/5X,'HYBRID MATRIX IN RADAR COORDINATES'/5X,'DIAGONAL TERMS
     1 ARE STANDARD DEVIATIONS, OFF-DIAGONAL TERMS ARE CORRELATION COEFF
     2ICIENTS'/(7D17.8))
C
C     TEST FOR MEMORY CLAMPING
C
 1302 IF(KLAMP)1055,1055,119
  119 IF(NN-KLAMP)1055,1193,1194
 1193 DELAST=DET
      GO TO 1055
C
C     SCALE COVARIANCE MATRIX
C
 1194 SCALE=(DELAST/DET)**.142857143
      DO 1195 J=1,7
      DO 1195 K=1,7
 1195 SI(J,K,NK)=SCALE*SI(J,K,NK)
 1000 CONTINUE
 1055 IF(NSTART)100,1200,100
 1200 RETURN
  500 WRITE (6,501)
  501 FORMAT(//5X,'*** MDATA =0, THIS SUBROUTINE SHOULD NOT HAVE BEEN CA
     1LLED ***'///)
      RETURN
      END
```

TABLE B-VI

SUBROUTINE DENS

```
      SUBROUTINE DENS
      COMMON /FCOM/COORD,DLAT,PRNT
      REAL*8 HKMFT,RHO,COORD,DLAT,PRNT
      DIMENSION HTKM(100),RHOM(100),HTKFT(100),RHOE(100)
C
C     SET UP ALTITUDE ARRAY HTKM, IN KILOMETERS
C
      DO 3 I=1,51
      HTKM(I)=2.0*FLOAT(I-1)
    3 CONTINUE
      DO 4 J=1,10
      HTKM(J+51)=110.0+FLOAT(J-1)*10.0
    4 CONTINUE
C
C     SET UP ATMOSPHERIC DENSITY ARRAY RHOM, IN KG/METER**3
C
      RHOM(1)=1.2250
      RHOM(2)=1.0066
      RHOM(3)=8.1935E-1
      RHOM(4)=6.6011E-1
      RHOM(5)=5.2579E-1
      RHOM(6)=4.1351E-1
      RHOM(7)=3.1194E-1
      RHOM(8)=2.2786E-1
      RHOM(9)=1.6647E-1
      RHOM(10)=1.2165E-1
      RHOM(11)=8.8910E-2
      RHOM(12)=6.4510E-2
      RHOM(13)=4.6938E-2
      RHOM(14)=3.4257E-2
      RHOM(15)=2.5076E-2
      RHOM(16)=1.8410E-2
      RHOM(17)=1.3555E-2
      RHOM(18)=9.8874E-3
      RHOM(19)=7.2579E-3
      RHOM(20)=5.3666E-3
      RHOM(21)=3.9957E-3
      RHOM(22)=2.9948E-3
      RHOM(23)=2.2589E-3
      RHOM(24)=1.7141E-3
      RHOM(25)=1.3167E-3
      RHOM(26)=1.0269E-3
      RHOM(27)=8.0097E-4
      RHOM(28)=6.3137E-4
      RHOM(29)=4.9762E-4
      RHOM(30)=3.9086E-4
      RHOM(31)=3.0592E-4
      RHOM(32)=2.3931E-4
      RHOM(33)=1.8837E-4
      RHOM(34)=1.4713E-4
      RHOM(35)=1.1399E-4
      RHOM(36)=8.7535E-5
      RHOM(37)=6.6593E-5
      RHOM(38)=5.0151E-5
      RHOM(39)=3.736E-5
      RHOM(40)=2.750E-5
      RHOM(41)=1.959E-5
      RHOM(42)=1.382E-5
      RHOM(43)=9.563E-6
      RHOM(44)=6.617E-6
```

TABLE B-VI (Continued)

```
      RHCM(45)=4.579E-6
      RHCM(46)=3.170E-6
      RHCM(47)=2.137E-6
      RHCM(48)=1.459E-6
      RHCM(49)=1.008E-6
      RHCM(50)=7.044E-7
      RHCM(51)=4.974E-7
      RHCM(52)=9.829E-8
      RHCM(53)=2.436E-8
      RHCM(54)=7.589E-9
      RHCM(55)=3.394E-9
      RHCM(56)=1.836E-9
      RHCM(57)=1.159E-9
      RHCM(58)=8.036E-10
      RHCM(59)=5.858E-10
      RHCM(60)=4.347E-10
      RHCM(61)=3.318E-10
C
C     SET UP ALTITUDE ARRAY HTKFT, IN KFT
C
      DO 5 I=1,51
      HTKFT(I)=6.0*FLOAT(I-1)
    5 CONTINUE
      DO 6 J=1,10
      HTKFT(J+51)=330.0+FLOAT(J-1)*30.0
    6 CONTINUE
C
C     SET UP ATMOSPHERIC DENSITY ARRAY,RHOE, IN LBS/FT**3
C
      RHOE(1)=7.6474E-2
      RHOE(2)=6.3925E-2
      RHOE(3)=5.3022E-2
      RHOE(4)=4.3606E-2
      RHOE(5)=3.5531E-2
      RHOE(6)=2.8657E-2
      RHOE(7)=2.2853E-2
      RHOE(8)=1.7170E-2
      RHOE(9)=1.2884E-2
      RHOE(10)=9.6701E-3
      RHOE(11)=7.2589E-3
      RHOE(12)=5.4485E-3
      RHOE(13)=4.0624E-3
      RHOE(14)=3.0368E-3
      RHOE(15)=2.2759E-3
      RHOE(16)=1.7100E-3
      RHOE(17)=1.2879E-3
      RHOE(18)=9.7241E-4
      RHOE(19)=7.3188E-4
      RHOE(20)=5.4944E-4
      RHOE(21)=4.5106E-4
      RHOE(22)=3.1543E-4
      RHOE(23)=2.4110E-4
      RHOE(24)=1.8530E-4
      RHOE(25)=1.4317E-4
      RHOE(26)=1.1119E-4
      RHOE(27)=8.6943E-5
      RHOE(28)=6.9261E-5
      RHOE(29)=5.5183E-5
      RHOE(30)=4.4159E-5
      RHOE(31)=3.5578E-5
```

TABLE B-VI (Continued)

```
      RHCE(32)=2.8583E-5
      RHCE(33)=2.2898E-5
      RHCE(34)=1.8289E-5
      RHCE(35)=1.4627E-5
      RHCE(36)=1.1748E-5
      RHCE(37)=9.3752E-6
      RHCE(38)=7.4307E-6
      RHCE(39)=5.8469E-6
      RHCE(40)=4.5653E-6
      RHCE(41)=3.5353E-6
      RHCE(42)=2.714E-6
      RHCE(43)=2.063E-6
      RHCE(44)=1.553E-6
      RHCE(45)=1.145E-6
      RHCE(46)=8.172E-7
      RHCE(47)=5.835E-7
      RHCE(48)=4.166E-7
      RHCE(49)=2.976E-7
      RHCE(50)=2.126E-7
      RHCE(51)=1.488E-7
      RHCE(52)=2.796E-8
      RHCE(53)=6.385E-9
      RHCE(54)=1.743E-9
      RHCE(55)=5.751E-10
      RHCE(56)=2.600E-10
      RHCE(57)=1.415E-10
      RHCE(58)=8.845E-11
      RHCE(59)=6.090E-11
      RHCE(60)=4.448E-11
      RHCE(61)=3.350E-11
      RETURN
C
      ENTRY ATM(HKMFT,RHO)
C
      RHO=0.0
      IF(HKMFT)13,100,100
  100 IF(CCORD)1,2,2
C
C     DATA IN METRIC UNITS
C
    1 IF(HKMFT-1.D2)11,11,15
   11 L=HKMFT/2.D0+1.D0
 12   A=(HKMFT-HTKM(L))/(HTKM(L+1)-HTKM(L))
 14   RHO=RHOM(L)*(RHOM(L+1)/RHOM(L))**A
      RETURN
   15 IF(HKMFT-2.D2)17,18,16
   17 L=51.000001+(HKMFT-1.D2)/1.D1
      GO TO 12
   18 RHO=RHOM(61)
      RETURN
C
C     DATA IN ENGLISH UNITS
C
    2 IF(HKMFT-3.D2)21,21,25
   21 L=HKMFT/6.D0+1.0000001
 22   A=(HKMFT-HTKFT(L))/(HTKFT(L+1)-HTKFT(L))
 24   RHO=RHOE(L)*(RHOE(L+1)/RHOE(L))**A
      RETURN
   25 IF(HKMFT-6.D2)26,27,16
   26 L=51.000001+(HKMFT-3.D2)/3.D1
```

TABLE B-VI (Continued)

```
      GO TO 22
   27 RHO=RHOE(61)
   16 RETURN
   13 WRITE(6,130)
  130 FORMAT(/5X,'***** EARTH IMPACT *****')
      RETURN
      END
```

TABLE B-VII
SUBROUTINE XYTOR

```
      SUBROUTINE XYTOR(X,Y,Z,XDOT,YDOT,ZDOT,R,A,E,RDOT,ADOT,EDOT)
C
C     SUBROUTINE TO CONVERT FROM XYZ COORDINATES TO RADAR POLAR
C     COORDINATES
C
      REAL*8 X,Y,Z,XDOT,YDOT,ZDOT,R,A,E,RDOT,ADOT,EDOT
      IMPLICIT REAL*8(A-H,O-Z)
      R=DSQRT(X**2+Y**2+Z**2)
      E=DARSIN(Z/R)
      A=DATAN2(X,Y)
      RDOT=(X*XDOT+Y*YDOT+Z*ZDOT)/R
      ADOT=(Y*XDOT-X*YDOT)/(X**2+Y**2)
      DENOM=DABS(R**2-Z**2)
      EDOT=(R*ZDOT-Z*RDOT)/(DSQRT(DENOM)*R)
      RETURN
      END
```

TABLE B-VIII
SUBROUTINE RTOXY

```
      SUBROUTINE RTOXY(R,A,E,RDOT,ADOT,EDOT,X,Y,Z,XDOT,YDOT,ZDOT)
C
C     SUBROUTINE TO CONVERT FROM RADAR COORDINATES TO XYZ COORDINATES
C
      REAL*8 R,A,E,RDOT,ADOT,EDOT,X,Y,Z,XDOT,YDOT,ZDOT
      IMPLICIT REAL*8(A-H,O-Z)
      SAZ=DSIN(A)
      SEL=DSIN(E)
      CAZ=DCOS(A)
      CEL=DCOS(E)
      X=R*CEL*SAZ
      Y=R*CEL*CAZ
      Z=R*SEL
      RPR=RDOT/R
      XDOT=RPR*X-EDOT*Z*SAZ+ADOT*Y
      YDOT=RPR*Y-EDOT*Z*CAZ-ADOT*X
      ZDOT=RPR*Z+EDOT*R*CEL
      RETURN
      END
```

TABLE B-IX

SUBROUTINE XCOVTR

```
      SUBROUTINE XCOVTR(XCOV,XVCTR,RCOV,NCODE)
      REAL*8 XCOV,XVCTR,RCOV
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION XVCTR(7)
      DIMENSION XCOV(7,7),RCOV(7,7),RVCTR(7)
      DIMENSION CONVT(7,7),DUM1(7,7)
      DIMENSION CONV(7,7)
      X =XVCTR(1)
      Y =XVCTR(2)
      Z =XVCTR(3)
      XP=XVCTR(4)
      YP=XVCTR(5)
      ZP=XVCTR(6)
      RVCTR(7)=XVCTR(7)
      CALL XYTOR(X,Y,Z,XP,YP,ZP,RVCTR(1),RVCTR(2),RVCTR(3),RVCTR(4),RVCT
     1R(5),RVCTR(6))
      RVSQ=RVCTR(1)**2
      RSQ=X**2+Y**2
      RXY=DSQRT(RSQ)
      DO 1 J=1,7
      DO 1 K=1,7
      RCOV(J,K)=0.
    1 CONV(J,K)=0.
      CONV(1,1)=X/RVCTR(1)
      CONV(1,2)=Y/RVCTR(1)
      CONV(2,1)=Y/RSQ
      CONV(2,2)=-X/RSQ
      CONV(1,3)=Z/RVCTR(1)
      CONV(3,1)=-X*Z/(RXY*RVSQ)
      CONV(3,2)=-Y*Z/(RXY*RVSQ)
      CONV(3,3)= RXY/RVSQ
      CONV(4,1)=(RVCTR(1)*XP-RVCTR(4)*X)/RVSQ
      CONV(4,2)=(RVCTR(1)*YP-RVCTR(4)*Y)/RVSQ
      CONV(4,3)=(RVCTR(1)*ZP-RVCTR(4)*Z)/RVSQ
      CONV(5,1)=-(2.DO*X*RVCTR(5)+YP)/RSQ
      CONV(5,2)=(XP-2.DO*Y*RVCTR(5))/RSQ
      CONV(6,1)=-X*RVCTR(6)/RSQ-2.DO*RVCTR(4)*CONV(3,1)/RVCTR(1)-Z*XP/(R
     1VSQ*RXY)
      CONV(6,2)=-Y*RVCTR(6)/RSQ-2.DO*RVCTR(4)*CONV(3,2)/RVCTR(1)-Z*YP/(R
     1VSQ*RXY)
      CONV(6,3)=-Z*RVCTR(6)/RVSQ-RVCTR(4)*CONV(3,3)/RVCTR(1)
      CONV(7,7)=1.
      DO 2 J=4,6
      DO 2 K=4,6
    2 CONV(J,K)=CONV(J-3,K-3)
      IF(NCODE)100,20,20
   20 CONTINUE
      DO 3 J=1,7
      DO 3 K=1,7
    3 CONVT(J,K)=CONV(K,J)
      CALL DMTMUL(XCOV,CONVT,DUM1,7,7,7)
      CALL DMTMUL(CONV,DUM1,RCOV, 7,7,7)
      RETURN
  100 CONTINUE
      DO 101 J=1,7
      DO 101 K=1,7
  101 RCOV(J,K)=CONV(J,K)
      RETURN
      END
```

TABLE B-X

ROUTINE RCOVTX

```
      SUBROUTINE RCOVTX(COVR,XSTATE,COVX)
      REAL*8 COVR,ESTATE,XSTATE,COVX
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION COVR(7,7),COVX(7,7),ESTATE(7),DXDR(7,7),DUMMY(7,7)
      DIMENSION XSTATE(7)
      CALL XYTOR(XSTATE(1),XSTATE(2),XSTATE(3),XSTATE(4),XSTATE(5),XSTAT
     1E(6),ESTATE(1),ESTATE(2),ESTATE(3),ESTATE(4),ESTATE(5),ESTATE(6))
      ESTATE(7)=XSTATE(7)
      DO 200 J=1,7
      DO 200 K=1,7
  200 DXDR(J,K)=0.
      SAZ=DSIN(ESTATE(2))
      CAZ=DCOS(ESTATE(2))
      SEL=DSIN(ESTATE(3))
      CEL=DCOS(ESTATE(3))
C
C     CONVERT COVARIANCE MATRIX TO XYZ COORDINATES
      DXDR(1,1)=CEL*SAZ
      DXDR(1,2)=ESTATE(1)*CEL*CAZ
      DXDR(1,3)=-ESTATE(1)*SEL*SAZ
      DXDR(2,1)=CEL*CAZ
      DXDR(2,2)=-ESTATE(1)*CEL*SAZ
      DXDR(2,3)=-ESTATE(1)*SEL*CAZ
      DXDR(3,1)=SEL
      DXDR(3,2)=0.
      DXDR(3,3)=ESTATE(1)*CEL
      DXDR(4,1)=ESTATE(5)*CEL*CAZ-ESTATE(6)*SEL*SAZ
      DXDR(4,2)=ESTATE(4)*CEL*CAZ-ESTATE(6)*ESTATE(1)*SEL*CAZ-ESTATE(5)*
     1ESTATE(1)*CEL*SAZ
      DXDR(4,3)=-ESTATE(4)*SEL*SAZ-ESTATE(6)*ESTATE(1)*CEL*SAZ-ESTATE(5)
     1*ESTATE(1)*SEL*CAZ
      DXDR(5,1)=-ESTATE(6)*SEL*CAZ-ESTATE(5)*CEL*SAZ
      DXDR(5,2)=-ESTATE(4)*CEL*SAZ+ESTATE(6)*ESTATE(1)*SEL*SAZ-ESTATE(5)
     1*ESTATE(1)*CEL*CAZ
      DXDR(5,3)=-ESTATE(4)*SEL*CAZ-ESTATE(6)*ESTATE(1)*CEL*CAZ+ESTATE(5)
     1*ESTATE(1)*SEL*SAZ
      DXDR(6,1)=ESTATE(6)*CEL
      DXDR(6,2)=0.
      DXDR(6,3)=ESTATE(4)*CEL-ESTATE(6)*ESTATE(1)*SEL
      DO 300 J=4,6
      DO 300 K=4,6
  300 DXDR(J,K)=DXDR(J-3,K-3)
      DXDR(7,7)=1.
      DO 301 J=1,7
      DO 301 K=1,7
      DUMMY(J,K)=0.
      DO 301 L=1,7
  301 DUMMY(J,K)=DUMMY(J,K)+COVR(J,L)*DXDR(K,L)
      DO 302 J=1,7
      DO 302 K=1,7
      COVX(J,K)=0.
      DO 302 L=1,7
  302 COVX(J,K)=COVX(J,K)+DXDR(J,L)*DUMMY(L,K)
      RETURN
      END
```

TABLE B-XI

SUBROUTINE GAUSSN

```
      SUBROUTINE GAUSSN(NS,RN,SIGMA)
      REAL*8 RN,SIGMA
      DIMENSION RN(1)
      DO 9 J=1,NS
    3 V=RAN(X)
      Y=-ALOG(V)
      V=RAN(X)
      Z=-ALOG(V)
      IF((Y-1.)**2-2.*Z)6,6,3
    6 S=RAN(X)
      IF(S-.5)7,6,8
    7 RN(J)=Y*SIGMA
      GO TO 9
    8 RN(J)=-Y*SIGMA
    9 CONTINUE
      RETURN
      END
```

TABLE B-XII

SUBROUTINE DMTMUL

```
      SUBROUTINE DMTMUL(A,B,AB,NRA,NCA,NCB)
      DOUBLE PRECISION A,B,AB
      DIMENSION A(7,7),B(7,7),AB(7,7)
      DO 1 J=1,NRA
      DO 1 K=1,NCB
      AB(J,K)=0.
      DO 1 L=1,NCA
    1 AB(J,K)=AB(J,K)+A(J,L)*B(L,K)
      RETURN
      END
```

TABLE B-XIII

SUBROUTINE MINV

```
C                                                                       MINV 001
C     ...................................................................MINV 002
C                                                                       MINV 003
C        SUBROUTINE MINV                                                MINV 004
C                                                                       MINV 005
C        PURPOSE                                                        MINV 006
C           INVERT A MATRIX                                             MINV 007
C                                                                       MINV 008
C        USAGE                                                          MINV 009
C           CALL MINV(A,N,D,L,M)                                        MINV 010
C                                                                       MINV 011
C        DESCRIPTION OF PARAMETERS                                      MINV 012
C           A - INPUT MATRIX, DESTROYED IN COMPUTATION AND REPLACED BY  MINV 013
C               RESULTANT INVERSE.                                      MINV 014
C           N - ORDER OF MATRIX A                                       MINV 015
C           D - RESULTANT DETERMINANT                                   MINV 016
C           L - WORK VECTOR OF LENGTH N                                 MINV 017
C           M - WORK VECTOR OF LENGTH N                                 MINV 018
C                                                                       MINV 019
C        REMARKS                                                        MINV 020
C           MATRIX A MUST BE A GENERAL MATRIX                           MINV 021
C                                                                       MINV 022
C        SUBROUTINES AND FUNCTION SUBPROGRAMS REQUIRED                  MINV 023
C           NONE                                                        MINV 024
C                                                                       MINV 025
C        METHOD                                                         MINV 026
C           THE STANDARD GAUSS-JORDAN METHOD IS USED. THE DETERMINANT   MINV 027
C           IS ALSO CALCULATED. A DETERMINANT OF ZERO INDICATES THAT    MINV 028
C           THE MATRIX IS SINGULAR.                                     MINV 029
C                                                                       MINV 030
C     ...................................................................MINV 031
C                                                                       MINV 032
      SUBROUTINE MINV(A,N,D,L,M)                                        MINV 033
      DIMENSION A(1),L(1),M(1)                                          MINV 034
C                                                                       MINV 035
C        .................................................................MINV 036
C                                                                       MINV 037
C        IF A DOUBLE PRECISION VERSION OF THIS ROUTINE IS DESIRED, THE  MINV 038
C        C IN COLUMN 1 SHOULD BE REMOVED FROM THE DOUBLE PRECISION      MINV 039
C        STATEMENT WHICH FOLLOWS.                                       MINV 040
C                                                                       MINV 041
      DOUBLE PRECISION A,D,BIGA,HOLD                                    MINV 042
C                                                                       MINV 043
C        THE C MUST ALSO BE REMOVED FROM DOUBLE PRECISION STATEMENTS    MINV 044
C        APPEARING IN OTHER ROUTINES USED IN CONJUNCTION WITH THIS      MINV 045
C        ROUTINE.                                                       MINV 046
C                                                                       MINV 047
C        THE DOUBLE PRECISION VERSION OF THIS SUBROUTINE MUST ALSO      MINV 048
C        CONTAIN DOUBLE PRECISION FORTRAN FUNCTIONS.  ABS IN STATEMENT  MINV 049
C        10 MUST BE CHANGED TO DABS.                                    MINV 050
C                                                                       MINV 051
C        .................................................................MINV 052
C                                                                       MINV 053
C        SEARCH FOR LARGEST ELEMENT                                     MINV 054
C                                                                       MINV 055
      D=1.0                                                             MINV 056
      NK=-N                                                             MINV 057
      DO 80 K=1,N                                                       MINV 058
      NK=NK+N                                                           MINV 059
      L(K)=K                                                            MINV 060
      M(K)=K                                                            MINV 061
```

TABLE B-XIII (Continued)

```
      KK=NK&K                                                          MINV 062
      BIGA=A(KK)                                                       MINV 063
      DO 20 J=K,N                                                      MINV 064
      IZ=N*(J-1)                                                       MINV 065
      DO 20 I=K,N                                                      MINV 066
      IJ=IZ&I                                                          MINV 067
   10 IF(DABS(BIGA)-DABS(A(IJ))) 15,20,20                              MINV 068
   15 BIGA=A(IJ)                                                       MINV 069
      L(K)=I                                                           MINV 070
      M(K)=J                                                           MINV 071
   20 CONTINUE                                                         MINV 072
C                                                                      MINV 073
C        INTERCHANGE ROWS                                              MINV 074
C                                                                      MINV 075
      J=L(K)                                                           MINV 076
      IF(J-K) 35,35,25                                                 MINV 077
   25 KI=K-N                                                           MINV 078
      DO 30 I=1,N                                                      MINV 079
      KI=KI&N                                                          MINV 080
      HOLD=-A(KI)                                                      MINV 081
      JI=KI-K&J                                                        MINV 082
      A(KI)=A(JI)                                                      MINV 083
   30 A(JI) =HOLD                                                      MINV 084
C                                                                      MINV 085
C        INTERCHANGE COLUMNS                                           MINV 086
C                                                                      MINV 087
   35 I=M(K)                                                           MINV 088
      IF(I-K) 45,45,38                                                 MINV 089
   38 JP=N*(I-1)                                                       MINV 090
      DO 40 J=1,N                                                      MINV 091
      JK=NK&J                                                          MINV 092
      JI=JP&J                                                          MINV 093
      HOLD=-A(JK)                                                      MINV 094
      A(JK)=A(JI)                                                      MINV 095
   40 A(JI) =HOLD                                                      MINV 096
C                                                                      MINV 097
C        DIVIDE COLUMN BY MINUS PIVOT (VALUE OF PIVOT ELEMENT IS       MINV 098
C        CONTAINED IN BIGA)                                            MINV 099
C                                                                      MINV 100
   45 IF(BIGA) 48,46,48                                                MINV 101
   46 D=0.0                                                            MINV 102
      RETURN                                                           MINV 103
   48 DO 55 I=1,N                                                      MINV 104
      IF(I-K) 50,55,50                                                 MINV 105
   50 IK=NK&I                                                          MINV 106
      A(IK)=A(IK)/(-BIGA)                                              MINV 107
   55 CONTINUE                                                         MINV 108
C                                                                      MINV 109
C        REDUCE MATRIX                                                 MINV 110
C                                                                      MINV 111
      DO 65 I=1,N                                                      MINV 112
      IK=NK&I                                                          MINV 113
      IJ=I-N                                                           MINV 114
      DO 65 J=1,N                                                      MINV 115
      IJ=IJ&N                                                          MINV 116
      IF(I-K) 60,65,60                                                 MINV 117
   60 IF(J-K) 62,65,62                                                 MINV 118
   62 KJ=IJ-I&K                                                        MINV 119
      A(IJ)=A(IK)*A(KJ)&A(IJ)                                          MINV 120
   65 CONTINUE                                                         MINV 121
```

TABLE B-XIII (Continued)

```
C                                                                    MINV 122
C         DIVIDE ROW BY PIVOT                                        MINV 123
C                                                                    MINV 124
      KJ=K-N                                                         MINV 125
      DO 75 J=1,N                                                    MINV 126
      KJ=KJ+N                                                        MINV 127
      IF(J-K) 70,75,70                                               MINV 128
   70 A(KJ)=A(KJ)/BIGA                                               MINV 129
   75 CONTINUE                                                       MINV 130
C                                                                    MINV 131
C         PRODUCT OF PIVOTS                                          MINV 132
C                                                                    MINV 133
      D=D*BIGA                                                       MINV 134
C                                                                    MINV 135
C         REPLACE PIVOT BY RECIPROCAL                                MINV 136
C                                                                    MINV 137
      A(KK)=1.0/BIGA                                                 MINV 138
   80 CONTINUE                                                       MINV 139
C                                                                    MINV 140
C         FINAL ROW AND COLUMN INTERCHANGE                           MINV 141
C                                                                    MINV 142
      K=N                                                            MINV 143
  100 K=(K-1)                                                        MINV 144
      IF(K) 150,150,105                                              MINV 145
  105 I=L(K)                                                         MINV 146
      IF(I-K) 120,120,108                                            MINV 147
  108 JQ=N*(K-1)                                                     MINV 148
      JR=N*(I-1)                                                     MINV 149
      DO 110 J=1,N                                                   MINV 150
      JK=JQ+J                                                        MINV 151
      HOLD=A(JK)                                                     MINV 152
      JI=JR+J                                                        MINV 153
      A(JK)=-A(JI)                                                   MINV 154
  110 A(JI) =HOLD                                                    MINV 155
  120 J=M(K)                                                         MINV 156
      IF(J-K) 100,100,125                                            MINV 157
  125 KI=K-N                                                         MINV 158
      DO 130 I=1,N                                                   MINV 159
      KI=KI+N                                                        MINV 160
      HOLD=A(KI)                                                     MINV 161
      JI=KI-K+J                                                      MINV 162
      A(KI)=-A(JI)                                                   MINV 163
  130 A(JI) =HOLD                                                    MINV 164
      GO TO 100                                                      MINV 165
  150 RETURN                                                         MINV 166
      END                                                            MINV 167
```

DOCUMENT CONTROL DATA – R&D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY *(Corporate author)* Lincoln Laboratory, M.I.T.	2a. REPORT SECURITY CLASSIFICATION Unclassified 2b. GROUP None

3. REPORT TITLE

A Trajectory Analysis Program (TRAP)

4. DESCRIPTIVE NOTES *(Type of report and inclusive dates)*

Technical Note

5. AUTHOR(S) *(Last name, first name, initial)*

Bertolini, Anthony

6. REPORT DATE	7a. TOTAL NO. OF PAGES	7b. NO. OF REFS
29 September 1967	86	6

8a. CONTRACT OR GRANT NO.	9a. ORIGINATOR'S REPORT NUMBER(S)
AF 19(628)-5167 b. PROJECT NO. ARPA Order 498 c. d.	Technical Note 1967-48 9b. OTHER REPORT NO(S) *(Any other numbers that may be assigned this report)* ESD-TR-67-508

10. AVAILABILITY/LIMITATION NOTICES

This document has been approved for public release and sale; its distribution is unlimited.

11. SUPPLEMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY
None	Advanced, Research Projects Agency, Department of Defense

13. ABSTRACT

A FORTRAN IV program package has been written for the generation, analysis, and estimation of re-entry trajectories. The following functions may be performed by the program:

(1) Generation of simulated noiseless trajectory data by integrating the differential equations of motion, using a predictor-corrector method.

(2) Error analyses on trajectories generated by the program, in which the effects of hypothetical errors in the initial state and subsequent measurements may be studied.

(3) Estimation of state vector quantities, using a recursive Kalman-filter scheme, from

(a) Simulated noisy data generated by the program.

(b) Noisy radar measurements, supplied externally to the program.

14. KEY WORDS

trajectory analysis
computer program
program package
FORTRAN IV
re-entry trajectories